

Reading Document Type Definitions

In an ideal world, every markup language created with XML would come with copious documentation and examples showing you the exact meaning and use of every element and attribute. In practice, most DTD authors, like most programmers, consider documentation an unpleasant and unnecessary chore, one best left to tech writers if it's to be done at all. Not surprisingly, therefore, the DTD that contains sufficient documentation is the exception, not the rule. Consequently, it's important to learn to read raw DTDs written by others.

There's a second good reason for learning to read DTDs. When you read good DTDs, you can often learn tricks and techniques that you can use in your own DTDs. For example, no matter how much theory I may mumble about the proper use of parameter entities for common attribute lists in DTDs, nothing proves quite as effective for learning that as really digging into a DTD that uses the technique. Reading other designers' DTDs teaches you by example how you can design your own.

In this chapter, we'll pick apart the modularized DTD for XHTML from the W3C. This DTD is quite complex and relatively well written. By studying it closely, you can pick up a lot of good techniques for developing your own DTDs. We'll see what its designers did right, and a few things they did wrong (IMHO). We'll explore some different ways the same thing could have been accomplished, and the advantages and disadvantages of each. We will also look at some common tricks in XML DTDs and techniques for developing your own DTDs.



In This Chapter

The importance of reading DTDs

What is XHTML?

The structure of the XHTML DTDs

The XHTML Modules

The XHTML Entity Sets

Simplified Subset DTDs

Techniques to imitate



The Importance of Reading DTDs

Some XML applications are very precisely defined by standards documents. MathML is one such application. It's been the subject of several person-years of work by a dedicated committee with representatives from across the computer math industry. It's been through several levels of peer review, and the committee's been quite responsive to problems discovered both in the language and in the documentation of that language. Consequently, a full DTD is available accompanied by an extensive informal specification.

Other XML applications are not as well documented. Microsoft, more or less, completely created CDF, discussed in Chapter 21. CDF is documented informally on Microsoft's Site Builder Network in a set of poorly organized Web pages, but no current DTD is available. Microsoft will probably update and add to CDF, but exactly what the updates will be is more or less a mystery to everyone else in the industry.

CML, the Chemical Markup Language invented by Peter Murray-Rust, is hardly documented at all. It contains a DTD, but it leaves a lot to the imagination. For instance, CML contains a `bondArray` element, but the only information about the `bondArray` element is that it contains `CDATA`. There's no further description of what sort of data should appear in a `bondArray` element.

Other times, there may be both a DTD and a prose specification. Microsoft and Marimba's Open Software Description (OSD format) is one example. However, the problem with prose specifications is that they leave pieces out. For instance, the spec for OSD generally neglects to say how many of a given child element may appear in a parent element or in what order. The DTD makes that clear. Conversely, the DTD can't really say that a `SIZE` attribute is given in the format `KB-number`. That's left to the prose part of the specification.

**Note**

Actually, this sort of information could and should appear in a comment in the DTD. The XML processor alone can't validate against this restriction. That has to be left to a higher layer of processing. In any case, simple comments can make the DTD more intelligible for humans, if nothing else. Currently, OSD does not have a solid DTD.

These are all examples of more or less public XML applications. However, many corporations, government agencies, Web sites, and other organizations have internal, private XML applications they use for their own documents. These are even less likely to be well documented and well written than the public XML applications. As an XML specialist, you may well find yourself trying to reverse engineer a DTD originally written by someone long gone and grown primarily through accretion of new elements over several years.

Clearly, the more documentation you have for an XML application, and the better the documentation is written, the easier it will be to learn and use that application. However it's an unfortunate fact of life that documentation is often an afterthought. Often, the only thing you have to work with is a DTD. You're reduced to reading the DTD, trying to understand what it says, and writing test documents to validate to try to figure out what is and isn't permissible. Consequently, it's important to be able to read DTDs and transform them in your head to examples of permissible markup.

In this chapter, you'll explore the XHTML DTD from the W3C. This is actually one of the better documented DTDs I've seen. However, in this chapter I'm going to pretend that it isn't. Instead of reading the prose specification, read the actual DTD files. We'll explore the techniques you can use to understand those DTDs, even in the absence of a prose specification.

What Is XHTML?

XHTML is the W3C's effort to rewrite HTML as strict XML. This requires tightening up a lot of the looseness commonly associated with HTML. End tags are required for elements that normally omit them like `p` and `dt`. Empty elements like `hr` must end in `/>` instead of just `>`. Attribute values must be quoted. The names of all HTML elements and attributes are standardized in lowercase.

XHTML goes one step further than merely requiring HTML documents to be well-formed XML like that discussed in Chapter 6. It actually provides a DTD for HTML you can use to validate your HTML documents. In fact, it provides three:

- ♦ The XHTML strict DTD for new HTML documents
- ♦ The XHTML loose DTD for converted old HTML documents that still use deprecated tags like `applet`
- ♦ The XHTML frameset DTD for documents that use frames

You can use the one that best fits your site.

Why Validate HTML?

Valid documents aren't absolutely required for HTML, but they do make it much easier for browsers to properly understand and display documents. A valid HTML document is far more likely to render correctly and predictably across many different browsers than an invalid one.

Until recently, too much of the competition among browser vendors revolved around just how much broken HTML they could make sense of. For instance, Internet Explorer fills in a missing `</table>` end tag whereas Netscape Navigator

does not. Consequently, many pages on Microsoft's Web site (which were only tested in Internet Explorer) contained missing `</table>` tags and could not be viewed in Netscape Navigator. (I'll leave it to the reader to decide whether or not this was deliberate sabotage.) In any case, if Microsoft had required valid HTML on its Web site, this would not have happened.

It is extremely difficult for even the largest Web shops to test their pages against even a small fraction of the browsers that people actually use. Even testing the latest versions of both Netscape Navigator and Internet Explorer is more than some designers manage. While I won't argue that you shouldn't test your pages in as many versions of as many browsers as possible in an ideal world, the reality is that time and resources are finite. Validating HTML goes a long way toward ensuring that your pages render reasonably in a broad spectrum of browsers.

Modularization of XHTML Working Draft

This chapter covers the April 6, 1999 working draft of the Modularized XHTML specification, which is subject to change. The status of this version is, as given by the W3C:

This document is a working draft of the W3C's HTML Working Group. This working draft may be updated, replaced or rendered obsolete by other W3C documents at any time. It is inappropriate to use W3C Working Drafts as reference material or to cite them as other than "work in progress." This is work in progress and does not imply endorsement by the W3C membership.

This document has been produced as part of the W3C HTML Activity. The goals of the HTML Working Group (*members only*) are discussed in the HTML Working Group charter (*members only*).

Currently, the latest draft is from April 6, 1999. You can download this particular version from <http://www.w3.org/TR/1999/xhtml1-modularization-19990406>. That document contains many more details about XHTML and rewriting Web pages in XML-compliant HTML. The most recent version is available on the Web at <http://www.w3.org/TR/xhtml1-modularization>. This chapter focuses on reading the DTD for XHTML. The files I reproduce and discuss below are subject to the W3C Document Notice, reproduced in the sidebar.

The Structure of the XHTML DTDs

HTML is a fairly complex XML application. As noted above, XHTML documents can choose one of three DTDs. The three separate HTML DTDs discussed here are divided into about 40 different files and over 2,000 lines of code. These files are connected through parameter entities. By splitting the DTD into these different files, it's easier to understand the individual pieces. Furthermore, common pieces can be shared among the three different versions of the XHTML DTD: strict, loose, and frameset.

Document Notice

Copyright (c) 1995-1999 World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved.

<http://www.w3.org/Consortium/Legal/>

Documents on the W3C site are provided by the copyright holders under the following license. By obtaining, using and/or copying this document, or the W3C document from which this statement is linked, you (the licensee) agree that you have read, understood, and will comply with the following terms and conditions:

Permission to use, copy, and distribute the contents of this document, or the W3C document from which this statement is linked, in any medium for any purpose and without fee or royalty is hereby granted, provided that you include the following on ALL copies of the document, or portions thereof, that you use:

1. A link or URL to the original W3C document.
2. The pre-existing copyright notice of the original author, if it doesn't exist, a notice of the form: "Copyright (c) World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. <http://www.w3.org/Consortium/Legal/>." (Hypertext is preferred, but a textual representation is permitted.)
3. If it exists, the STATUS of the W3C document.

When space permits, inclusion of the full text of this NOTICE should be provided. We request that authorship attribution be provided in any software, documents, or other items or products that you create pursuant to the implementation of the contents of this document, or any portion thereof.

No right to create modifications or derivatives of W3C documents is granted pursuant to this license.

THIS DOCUMENT IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DOCUMENT ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE DOCUMENT OR THE PERFORMANCE OR IMPLEMENTATION OF THE CONTENTS THEREOF.

The name and trademarks of copyright holders may NOT be used in advertising or publicity pertaining to this document or its contents without specific, written prior permission. Title to copyright in this document will at all times remain with copyright holders.

The three DTDs that can be used by your HTML in XML documents are listed below:

1. The XHTML strict DTD for new HTML documents.
2. The XHTML loose DTD for converted old HTML documents that still use deprecated tags like `applet`.
3. The XHTML frameset DTD for documents that use frames.

All three of these DTDs have this basic format:

1. Comment with title, copyright, namespace, formal public identifier, and other information for people who use this DTD.
2. Revised parameter entity declarations that will override parameter entities declared in the modules.
3. External parameter entity references to import the modules and entity sets.

XHTML Strict DTD

The XHTML strict DTD (`XHTML1-s.dtd`), shown in Listing 20-1, is for new HTML documents that can easily conform to the most stringent requirements for XML compatibility, and that do not need to use some of the older, less-well thought out and deprecated elements from HTML like `applet` and `basefont`. It does not support frames, and omits support for all presentational elements like `font` and `center`.

Listing 20-1: XHTML1-s.dtd: the XHTML strict DTD

```
<!-- ..... ->
<!-- XHTML 1.0 Strict DTD ..... ->
<!-- file: XHTML1-s.dtd
->
```

```
<!-- XHTML 1.0 Strict DTD
```

```
    This is XHTML 1.0, an XML reformulation of HTML 4.0.
```

```
Copyright 1998-1999 World Wide Web Consortium
(Massachusetts Institute of Technology, Institut National de
Recherche en Informatique et en Automatique, Keio University).
All Rights Reserved.
```

```
Permission to use, copy, modify and distribute the XHTML
1.0 DTD and its accompanying documentation for any purpose
and without fee is hereby granted in perpetuity, provided
that the above copyright notice and this paragraph appear
in all copies. The copyright holders make no representation
```

about the suitability of the DTD for any purpose.

It is provided "as is" without expressed or implied warranty.

Author: Murray M. Altheim <altheim@eng.sun.com>
Revision: @(#)XHTML1-s.dtd 1.14 99/04/01 SMI

The XHTML 1.0 DTD is an XML variant based on the W3C HTML 4.0 DTD:

Draft: \$Date: 1999/04/02 14:27:27 \$

Authors: Dave Raggett <dsr@w3.org>
Arnaud Le Hors <lehors@w3.org>
Ian Jacobs <ij@w3.org>

→

<!-- This is the driver file for version 1.0 of the XHTML Strict DTD.

Please use this formal public identifier to identify it:

"-//W3C//DTD XHTML 1.0 Strict//EN"

Please use this URI to identify the default namespace:

"http://www.w3.org/TR/1999/REC-html-in-xml"

For example, if you are using XHTML 1.0 directly, use the FPI in the DOCTYPE declaration, with the xmlns attribute on the document element to identify the default namespace:

```
<?xml version="1.0" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "XHTML1-s.dtd" >
<html xmlns="http://www.w3.org/TR/1999/REC-html-in-xml"
    xml:lang="en" lang="en" >
...
</html>
```

→

<!-- The version attribute has historically been a container for the DTD's public identifier (an FPI), but is unused in Strict: →

```
<!ENTITY % HTML.version "" >
<!ENTITY % Version.attrib "" >
```

<!-- The xmlns attribute on <html> identifies the default namespace to namespace-aware applications: →

```
<!ENTITY % XHTML.ns "http://www.w3.org/TR/1999/REC-html-in-xml" >
```

Continued

Listing 20-1 (continued)

```

<!-- reserved for future use with document profiles -->
<!ENTITY % XHTML.profile "" >

<!-- used to ignore Transitional features within modules -->
<!ENTITY % XHTML.Transitional "IGNORE" >

<!-- XHTML Base Architecture Module (optional) ..... -->
<!ENTITY % XHTML1-arch.module "IGNORE" >
<![%XHTML1-arch.module;[
<!ENTITY % XHTML1-arch.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Base Architecture//EN"
        "XHTML1-arch.mod" >
%XHTML1-arch.mod;
]]>

<!-- Common Names Module ..... -->
<!ENTITY % XHTML1-names.module "INCLUDE" >
<![%XHTML1-names.module;[
<!ENTITY % XHTML1-names.mod
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Common Names//EN"
        "XHTML1-names.mod" >
%XHTML1-names.mod;
]]>

<!-- Character Entities Module ..... -->
<!ENTITY % XHTML1-charent.module "INCLUDE" >
<![%XHTML1-charent.module;[
<!ENTITY % XHTML1-charent.mod
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Character Entities//EN"
        "XHTML1-charent.mod" >
%XHTML1-charent.mod;
]]>

<!-- Intrinsic Events Module ..... -->
<!ENTITY % XHTML1-events.module "INCLUDE" >
<![%XHTML1-events.module;[
<!ENTITY % XHTML1-events.mod
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Intrinsic Events//EN"
        "XHTML1-events.mod" >
%XHTML1-events.mod;
]]>

<!-- Common Attributes Module ..... -->
<!ENTITY % XHTML1-attribs.module "INCLUDE" >
<![%XHTML1-attribs.module;[
<!ENTITY % align "" >
<!ENTITY % XHTML1-attribs.mod
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Common Attributes//EN"
        "XHTML1-attribs.mod" >

```

```

%XHTML1-attrs.mod;
]]>

<!-- Document Model Module ..... ->
<!ENTITY % XHTML1-model.module "INCLUDE" >
<![%XHTML1-model.module;[
<!ENTITY % XHTML1-model.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Document Model//EN"
        "XHTML1-model.mod" >
%XHTML1-model.mod;
]]>

<!-- Inline Structural Module ..... ->
<!ENTITY % XHTML1-inlstruct.module "INCLUDE" >
<![%XHTML1-inlstruct.module;[
<!ENTITY % XHTML1-inlstruct.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Structural//EN"
        "XHTML1-inlstruct.mod" >
%XHTML1-inlstruct.mod;
]]>

<!-- Inline Presentational Module ..... ->
<!ENTITY % XHTML1-inlpres.module "INCLUDE" >
<![%XHTML1-inlpres.module;[
<!ENTITY % XHTML1-inlpres.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Presentational//EN"
        "XHTML1-inlpres.mod" >
%XHTML1-inlpres.mod;
]]>

<!-- Inline Phrasal Module ..... ->
<!ENTITY % XHTML1-inlphras.module "INCLUDE" >
<![%XHTML1-inlphras.module;[
<!ENTITY % XHTML1-inlphras.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Phrasal//EN"
        "XHTML1-inlphras.mod" >
%XHTML1-inlphras.mod;
]]>

<!-- Block Structural Module ..... ->
<!ENTITY % XHTML1-blkstruct.module "INCLUDE" >
<![%XHTML1-blkstruct.module;[
<!ENTITY % XHTML1-blkstruct.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Structural//EN"
        "XHTML1-blkstruct.mod" >
%XHTML1-blkstruct.mod;
]]>

<!-- Block Presentational Module ..... ->
<!ENTITY % XHTML1-blkpres.module "INCLUDE" >
<![%XHTML1-blkpres.module;[
<!ENTITY % XHTML1-blkpres.mod

```

Continued

Listing 20-1 (continued)

```

        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block
Presentational//EN"
        "XHTML1-blkpres.mod" >
%XHTML1-blkpres.mod;
]]>

<!-- Block Phrasal Module ..... ->
<!ENTITY % XHTML1-blkphras.module "INCLUDE" >
<![%XHTML1-blkphras.module;[
<!ENTITY % XHTML1-blkphras.mod
        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Phrasal//EN"
        "XHTML1-blkphras.mod" >
%XHTML1-blkphras.mod;
]]>

<!-- Scripting Module ..... ->
<!ENTITY % XHTML1-script.module "INCLUDE" >
<![%XHTML1-script.module;[
<!ENTITY % XHTML1-script.mod
        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Scripting//EN"
        "XHTML1-script.mod" >
%XHTML1-script.mod;
]]>

<!-- Stylesheets Module ..... ->
<!ENTITY % XHTML1-style.module "INCLUDE" >
<![%XHTML1-style.module;[
<!ENTITY % XHTML1-style.mod
        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Stylesheets//EN"
        "XHTML1-style.mod" >
%XHTML1-style.mod;
]]>

<!-- Image Module ..... ->
<!ENTITY % XHTML1-image.module "INCLUDE" >
<![%XHTML1-image.module;[
<!ENTITY % XHTML1-image.mod
        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Images//EN"
        "XHTML1-image.mod" >
%XHTML1-image.mod;
]]>

<!-- Frames Module ..... ->
<!ENTITY % XHTML1-frames.module "IGNORE" >
<![%XHTML1-frames.module;[
<!ENTITY % XHTML1-frames.mod
        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Frames//EN"
        "XHTML1-frames.mod" >
%XHTML1-frames.mod;
]]>

```

```

<!-- Linking Module ..... ->
<!ENTITY % XHTML1-linking.module "INCLUDE" >
<![%XHTML1-linking.module;[
<!ENTITY % XHTML1-linking.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Linking//EN"
        "XHTML1-linking.mod" >
%XHTML1-linking.mod;
]]>

<!-- Client-side Image Map Module ..... ->
<!ENTITY % XHTML1-csismap.module "INCLUDE" >
<![%XHTML1-csismap.module;[
<!ENTITY % XHTML1-csismap.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Client-side Image Map//EN"
        "XHTML1-csismap.mod" >
%XHTML1-csismap.mod;
]]>

<!-- Object Element Module ..... ->
<!ENTITY % XHTML1-object.module "INCLUDE" >
<![%XHTML1-object.module;[
<!ENTITY % XHTML1-object.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Object Element//EN"
        "XHTML1-object.mod" >
%XHTML1-object.mod;
]]>

<!-- Lists Module ..... ->
<!ENTITY % XHTML1-list.module "INCLUDE" >
<![%XHTML1-list.module;[
<!ENTITY % XHTML1-list.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Lists//EN"
        "XHTML1-list.mod" >
%XHTML1-list.mod;
]]>

<!-- Forms Module ..... ->
<!ENTITY % XHTML1-form.module "INCLUDE" >
<![%XHTML1-form.module;[
<!ENTITY % XHTML1-form.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Forms//EN"
        "XHTML1-form.mod" >
%XHTML1-form.mod;
]]>

<!-- Tables Module ..... ->
<!ENTITY % XHTML1-table.module "INCLUDE" >
<![%XHTML1-table.module;[
<!ENTITY % XHTML1-table.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Tables//EN"
        "XHTML1-table.mod" >
%XHTML1-table.mod;

```

Continued

Listing 20-1 (continued)

```

]]>

<!-- Document Metainformation Module ..... ->
<!ENTITY % XHTML1-meta.module "INCLUDE" >
<![%XHTML1-meta.module;[
<!ENTITY % XHTML1-meta.mod
      PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Metainformation//EN"
            "XHTML1-meta.mod" >
%XHTML1-meta.mod;
]]>

<!-- Document Structure Module ..... ->
<!ENTITY % XHTML1-struct.module "INCLUDE" >
<![%XHTML1-struct.module;[
<!ENTITY % XHTML1-struct.mod
      PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Document Structure//EN"
            "XHTML1-struct.mod" >
%XHTML1-struct.mod;
]]>

<!-- end of XHTML 1.0 Strict DTD ..... ->
<!-- ..... ->

```

The file begins with a comment identifying which file this is, and a basic copyright statement. That's followed by these very important words:

Permission to use, copy, modify, and distribute the XHTML 1.0 DTD and its accompanying documentation for any purpose and without fee is hereby granted in perpetuity, provided that the above copyright notice and this paragraph appear in all copies. The copyright holders make no representation about the suitability of the DTD for any purpose.

A statement like this is *very* important for any DTD that you want to be broadly adopted. In order for people outside your organization to use your DTD, they must be allowed to copy it, put it on their Web servers, send it to other people with their own documents, and do a variety of other things normally prohibited by copyright. A simple statement like "Copyright 1999 XYZ Corp." with no further elucidation prevents many people from using your DTD.

Next comes a comment containing detailed information about how this DTD should be used including its formal public identifier and preferred name. Also provided are the preferred namespace and an example of how to begin a file that uses this DTD. All of this is very useful to an author.



Formal public identifiers are discussed in Chapter 8, *Document Type Definitions and Validity*.

Next come several entity definitions that are mostly for compatibility with old or future versions of this DTD. Finally, we get to the meat of the DTD: 24 external parameter entity definitions and references that import the modules used to form the complete DTD. Here's the last one in the file:

```
<!-- Document Structure Module ..... ->
<!ENTITY % XHTML1-struct.module "INCLUDE" >
<![%XHTML1-struct.module;[
<!ENTITY % XHTML1-struct.mod
      PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Document Structure//EN"
           "XHTML1-struct.mod" >
%XHTML1-struct.mod;
]]>
```

All 24 follow the same basic structure:

1. A comment identifying the module to be imported.
2. A parameter entity declaration whose name is the name of the module to be imported suffixed with `.module` and whose replacement text is either `INCLUDE` or `IGNORE`.
3. An `INCLUDE` or `IGNORE` block; which one is determined by the value of the parameter entity reference in the previous step.
4. An external parameter entity declaration for the module to be imported suffixed with `.mod`, followed by an external parameter entity reference that actually imports the module.

Removing the module-specific material, the structure looks like this:

```
<!-- Module Name Module ..... ->
<!ENTITY % XHTML1-module_abbreviation.module "INCLUDE" >
<![%XHTML1-module_abbreviation.module;[
<!ENTITY % XHTML1-module_abbreviation.mod
      PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Module Name//EN"
           "XHTML1-module_abbreviation.mod" >
%XHTML1-module_abbreviation.mod;
]]>
```

The way this is organized it is very easy to change, whether or not a particular module is loaded simply by changing the value of one internal parameter entity from `INCLUDE` to `IGNORE` or vice versa. The `.module` parameter entities act as switches that turn particular declarations on or off.

XHTML Transitional DTD

The XHTML transitional DTD (`XHTML1-t.dtd`), also known as the loose DTD and shown in Listing 20-2, is appropriate for HTML documents that have not fully made the transition to HTML 4.0. These documents depend on now deprecated elements like `applet` and `center`. It also adds support for presentational attributes like `color` and `bullet` styles for list items replaced with CSS style sheets in strict HTML 4.0.

Listing 20-2: XHTML1-t.dtd: the XHTML transitional DTD

```

<!-- ..... ->
<!-- XHTML 1.0 Transitional DTD
..... ->
<!-- file: XHTML1-t.dtd
->

<!-- XHTML 1.0 Transitional DTD

    This is XHTML 1.0, an XML reformulation of HTML 4.0.

    Copyright 1998-1999 World Wide Web Consortium
    (Massachusetts Institute of Technology, Institut
    National de Recherche en Informatique et en
    Automatique, Keio University). All Rights Reserved.

    Permission to use, copy, modify and distribute the XHTML
    1.0 DTD and its accompanying documentation for any
    purpose and without fee is hereby granted in perpetuity,
    provided that the above copyright notice and this
    paragraph appear in all copies. The copyright holders
    make no representation about the suitability of the DTD
    for any purpose.

    It is provided "as is" without expressed or implied
    warranty.

    Author:      Murray M. Altheim <altheim@eng.sun.com>
    Revision:    @(#)XHTML1-t.dtd 1.14 99/04/01 SMI

    The XHTML 1.0 DTD is an XML variant based on the
    W3C HTML 4.0 DTD:

    Draft:      $Date: 1999/04/02 14:27:27 $

    Authors:    Dave Raggett <dsr@w3.org>
                Arnaud Le Hors <lehors@w3.org>
                Ian Jacobs <ij@w3.org>

->
<!-- This is the driver file for version 1.0 of the
XHTML Transitional DTD.

    Please use this formal public identifier to identify it:

        "-//W3C//DTD XHTML 1.0 Transitional//EN"

    Please use this URI to identify the default namespace:

        "http://www.w3.org/TR/1999/REC-html-in-xml"

    For example, if you are using XHTML 1.0 directly,

```

use the FPI in the DOCTYPE declaration, with the xmlns attribute on the document element to identify the default namespace:

```
<?xml version="1.0" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "XHTML1-t.dtd" >
<html xmlns="http://www.w3.org/TR/1999/REC-html-in-xml"
    xml:lang="en" lang="en" >
    ...
</html>
->
```

<!-- The version attribute has historically been a container for the DTD's public identifier (an FPI): ->

```
<!ENTITY % HTML.version "-//W3C//DTD XHTML 1.0
Transitional//EN" >
```

<!-- The xmlns attribute on <html> identifies the default namespace to namespace-aware applications: ->

```
<!ENTITY % XHTML.ns "http://www.w3.org/TR/1999/REC-html-in-
xml" >
```

<!-- reserved for future use with document profiles ->

```
<!ENTITY % XHTML.profile "" >
```

```
<!ENTITY % XHTML1-frames.module "IGNORE" >
```

```
<!ENTITY % XHTML.Transitional "INCLUDE" >
```

<!-- XHTML Base Architecture Module (optional) ->

```
<!ENTITY % XHTML1-arch.module "IGNORE" >
```

```
<![%XHTML1-arch.module;[
```

```
<!ENTITY % XHTML1-arch.mod
```

```
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Base Architecture//EN"
        "XHTML1-arch.mod" >
```

```
%XHTML1-arch.mod;
```

```
]]>
```

<!-- Common Names Module ->

```
<!ENTITY % XHTML1-names.module "INCLUDE" >
```

```
<![%XHTML1-names.module;[
```

```
<!ENTITY % XHTML1-names.mod
```

```
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Common Names//EN"
        "XHTML1-names.mod" >
```

```
%XHTML1-names.mod;
```

```
]]>
```

<!-- Character Entities Module ->

```
<!ENTITY % XHTML1-charent.module "INCLUDE" >
```

```
<![%XHTML1-charent.module;[
```

```
<!ENTITY % XHTML1-charent.mod
```

Continued

Listing 20-2 (continued)

```

        PUBLIC "-//W3C//ENTITIES XHTML 1.0 Character Entities//EN"
            "XHTML1-charent.mod" >
%XHTML1-charent.mod;
]]>

<!-- Intrinsic Events Module ..... ->
<!ENTITY % XHTML1-events.module "INCLUDE" >
<![%XHTML1-events.module;[
<!ENTITY % XHTML1-events.mod
        PUBLIC "-//W3C//ENTITIES XHTML 1.0 Intrinsic Events//EN"
            "XHTML1-events.mod" >
%XHTML1-events.mod;
]]>

<!-- Transitional Attributes Module ..... ->
<!ENTITY % XHTML1-attribs-t.module "INCLUDE" >
<![%XHTML1-attribs-t.module;[
<!ENTITY % XHTML1-attribs-t.mod
PUBLIC "-//W3C//ENTITIES XHTML 1.0 Transitional Attributes//EN"
        "XHTML1-attribs-t.mod" >
%XHTML1-attribs-t.mod;
]]>

<!-- Transitional Document Model Module ..... ->
<!ENTITY % XHTML1-model-t.module "INCLUDE" >
<![%XHTML1-model-t.module;[
<!ENTITY % XHTML1-model-t.mod
        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Transitional Document
Model//EN"
            "XHTML1-model-t.mod" >
%XHTML1-model-t.mod;
]]>

<!-- Inline Structural Module ..... ->
<!ENTITY % XHTML1-inlstruct.module "INCLUDE" >
<![%XHTML1-inlstruct.module;[
<!ENTITY % XHTML1-inlstruct.mod
        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Structural//EN"
            "XHTML1-inlstruct.mod" >
%XHTML1-inlstruct.mod;
]]>

<!-- Inline Presentational Module ..... ->
<!ENTITY % XHTML1-inlpres.module "INCLUDE" >
<![%XHTML1-inlpres.module;[
<!ENTITY % XHTML1-inlpres.mod
        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Presentational//EN"
            "XHTML1-inlpres.mod" >
%XHTML1-inlpres.mod;
]]>

```

```

<!-- Inline Phrasal Module ..... ->
<!ENTITY % XHTML1-inlphras.module "INCLUDE" >
<![%XHTML1-inlphras.module;[
<!ENTITY % XHTML1-inlphras.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Phrasal//EN"
        "XHTML1-inlphras.mod" >
%XHTML1-inlphras.mod;
]]>

<!-- Block Structural Module ..... ->
<!ENTITY % XHTML1-blkstruct.module "INCLUDE" >
<![%XHTML1-blkstruct.module;[
<!ENTITY % XHTML1-blkstruct.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Structural//EN"
        "XHTML1-blkstruct.mod" >
%XHTML1-blkstruct.mod;
]]>

<!-- Block Presentational Module ..... ->
<!ENTITY % XHTML1-blkpres.module "INCLUDE" >
<![%XHTML1-blkpres.module;[
<!ENTITY % XHTML1-blkpres.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Presentational//EN"
        "XHTML1-blkpres.mod" >
%XHTML1-blkpres.mod;
]]>

<!-- Block Phrasal Module ..... ->
<!ENTITY % XHTML1-blkphras.module "INCLUDE" >
<![%XHTML1-blkphras.module;[
<!ENTITY % XHTML1-blkphras.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Phrasal//EN"
        "XHTML1-blkphras.mod" >
%XHTML1-blkphras.mod;
]]>

<!-- Scripting Module ..... ->
<!ENTITY % XHTML1-script.module "INCLUDE" >
<![%XHTML1-script.module;[
<!ENTITY % XHTML1-script.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Scripting//EN"
        "XHTML1-script.mod" >
%XHTML1-script.mod;
]]>

<!-- Stylesheets Module ..... ->
<!ENTITY % XHTML1-style.module "INCLUDE" >
<![%XHTML1-style.module;[
<!ENTITY % XHTML1-style.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Stylesheets//EN"
        "XHTML1-style.mod" >
%XHTML1-style.mod;

```

Continued

Listing 20-2 (continued)

```

]]>

<!-- Image Module ..... ->
<!ENTITY % XHTML1-image.module "INCLUDE" >
<![%XHTML1-image.module;[
<!ENTITY % XHTML1-image.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Images//EN"
        "XHTML1-image.mod" >
%XHTML1-image.mod;
]]>

<!-- Frames Module ..... ->
<![%XHTML1-frames.module;[
<!ENTITY % XHTML1-frames.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Frames//EN"
        "XHTML1-frames.mod" >
%XHTML1-frames.mod;
]]>

<!-- Linking Module ..... ->
<!ENTITY % XHTML1-linking.module "INCLUDE" >
<![%XHTML1-linking.module;[
<!ENTITY % XHTML1-linking.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Linking//EN"
        "XHTML1-linking.mod" >
%XHTML1-linking.mod;
]]>

<!-- Client-side Image Map Module ..... ->
<!ENTITY % XHTML1-csismap.module "INCLUDE" >
<![%XHTML1-csismap.module;[
<!ENTITY % XHTML1-csismap.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Client-side Image Map//EN"
        "XHTML1-csismap.mod" >
%XHTML1-csismap.mod;
]]>

<!-- Object Element Module ..... ->
<!ENTITY % XHTML1-object.module "INCLUDE" >
<![%XHTML1-object.module;[
<!ENTITY % XHTML1-object.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Object Element//EN"
        "XHTML1-object.mod" >
%XHTML1-object.mod;
]]>

<!-- Java Applet Element Module ..... ->
<!ENTITY % XHTML1-applet.module "INCLUDE" >
<![%XHTML1-applet.module;[
<!ENTITY % XHTML1-applet.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Java Applets//EN"

```

```

        "XHTML1-applet.mod" >
%XHTML1-applet.mod;
]]>

<!-- Lists Module ..... ->
<!ENTITY % XHTML1-list.module "INCLUDE" >
<![%XHTML1-list.module;[
<!ENTITY % XHTML1-list.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Lists//EN"
        "XHTML1-list.mod" >
%XHTML1-list.mod;
]]>

<!-- Forms Module ..... ->
<!ENTITY % XHTML1-form.module "INCLUDE" >
<![%XHTML1-form.module;[
<!ENTITY % XHTML1-form.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Forms//EN"
        "XHTML1-form.mod" >
%XHTML1-form.mod;
]]>

<!-- Tables Module ..... ->
<!ENTITY % XHTML1-table.module "INCLUDE" >
<![%XHTML1-table.module;[
<!ENTITY % XHTML1-table.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Tables//EN"
        "XHTML1-table.mod" >
%XHTML1-table.mod;
]]>

<!-- Document Metainformation Module ..... ->
<!ENTITY % XHTML1-meta.module "INCLUDE" >
<![%XHTML1-meta.module;[
<!ENTITY % XHTML1-meta.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Metainformation//EN"
        "XHTML1-meta.mod" >
%XHTML1-meta.mod;
]]>

<!-- Document Structure Module ..... ->
<!ENTITY % XHTML1-struct.module "INCLUDE" >
<![%XHTML1-struct.module;[
<!ENTITY % XHTML1-struct.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Document Structure//EN"
        "XHTML1-struct.mod" >
%XHTML1-struct.mod;
]]>

<!-- end of XHTML 1.0 Transitional DTD
..... ->
<!-- ..... ->

```

This DTD is organized along the same lines as the strict DTD. First, comments tell you how to use this DTD. Next come entity declarations that are important for the imported modules, particularly `XHTML.Transitional` which is defined here as `INCLUDE`. In the strict DTD this was defined as `IGNORE`. Thus, the individual modules can use this to provide features that will only apply when the transitional DTD is being used. Finally, the various modules are imported. The difference between the strict and transitional DTDs lies in which modules are imported and how the parameter entities are overridden. The transitional DTD supports a superset of the strict DTD.

The XHTML Frameset DTD

The XHTML frameset DTD (`XHTML1-f.dtd`), shown in Listing 20-3, is a superset of the transitional DTD that adds support for frames.

Listing 20-3: XHTML1-f.dtd: the Voyager loose DTD for documents with frames

```

<!-- ..... ->
<!-- XHTML 1.0 Frameset DTD
..... ->
<!-- file: XHTML1-f.dtd
->

<!-- XHTML 1.0 Frameset DTD

This is XHTML 1.0, an XML reformulation of HTML 4.0.

Copyright 1998-1999 World Wide Web Consortium
(Massachusetts Institute of Technology, Institut
National de Recherche en Informatique et en
Automatique, Keio University). All Rights Reserved.

Permission to use, copy, modify and distribute the XHTML
1.0 DTD and its accompanying documentation for any
purpose and without fee is hereby granted in perpetuity,
provided that the above copyright notice and this
paragraph appear in all copies. The copyright holders
make no representation about the suitability of the DTD
for any purpose.

It is provided "as is" without expressed or implied
warranty.

Author: Murray M. Altheim <altheim@eng.sun.com>
Revision: @(#)XHTML1-f.dtd 1.17 99/04/01 SMI

The XHTML 1.0 DTD is an XML variant based on
the W3C HTML 4.0 DTD:

```

```

Draft:      $Date: 1999/04/02 14:27:26 $

Authors:    Dave Raggett <dsr@w3.org>
            Arnaud Le Hors <lehors@w3.org>
            Ian Jacobs <ij@w3.org>

->
<!-- This is the driver file for version 1.0 of
      the XHTML Frameset DTD.

      Please use this formal public identifier to identify it:

          "-//W3C//DTD XHTML 1.0 Frameset//EN"

      Please use this URI to identify the default namespace:

          "http://www.w3.org/TR/1999/REC-html-in-xml"

      For example, if you are using XHTML 1.0 directly, use the
      FPI in the DOCTYPE declaration, with the xmlns attribute
      on the document element to identify the default
      namespace:

<?xml version="1.0" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
          "XHTML1-f.dtd" >
<html xmlns="http://www.w3.org/TR/1999/REC-html-in-xml"
      xml:lang="en" lang="en" >
    ...
</html>
->

<!-- The version attribute has historically been a container
      for the DTD's public identifier (an FPI): ->
<!ENTITY % HTML.version "-//W3C//DTD XHTML 1.0 Frameset//EN" >

<!-- The xmlns attribute on <html> identifies the
      default namespace to namespace-aware applications: ->
<!ENTITY % XHTML.ns
      "http://www.w3.org/TR/1999/REC-html-in-xml" >

<!-- reserved for future use with document profiles ->
<!ENTITY % XHTML.profile "" >

<!ENTITY % XHTML1-frames.module "INCLUDE" >
<!ENTITY % XHTML.Transitional "INCLUDE" >

<!-- declare and instantiate the XHTML Transitional DTD ->
<!ENTITY % XHTML1-t.dtd
      PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

```

Continued

Listing 20-3 (continued)

```

        "XHTML1-t.dtd" >
%XHTML1-t.dtd;

<!-- end of XHTML 1.0 Frameset DTD
..... ->
<!-- ..... ->

```

This DTD is organized differently than the previous two DTDs. Instead of repeating all the definitions already in the transitional DTD, it simply imports that DTD using the XHTML1-t.dtd external parameter entity. Before doing this, however, it defines XHTML1-frames.module as INCLUDE. This entity was defined in the transitional DTD as IGNORE. However, the definition given here takes precedence. This DTD changes the meaning of the DTD it imports.

You could make a strict DTD that uses frames by importing the strict DTD instead of the transitional DTD like this:

```

<!-- declare and instantiate the XHTML Strict DTD ->
<!ENTITY % XHTML1-s.dtd
        PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "XHTML1-s.dtd" >
%XHTML1-s.dtd;

```

Other DTDs Although, XHTML1-s.dtd, XHTML1-t.dtd and XHTML1-f.dtd are the three main document types you can create with XHTML several other possibilities exist. One is documented in XHTML1-m.dtd, a DTD that includes both HTML and MathML (with a couple of changes needed to make MathML fully compatible with HTML).

There are also flat versions of the three main DTDs that use a single DTD file rather than many separate modules. They don't define different XML applications, and they're not as easy to follow as the modularized DTDs discussed here, but they are easier to place on Web sites. These include:

- ♦ XHTML1-s-flat.dtd: a strict XHTML DTD in a single file
- ♦ XHTML1-t-flat.dtd: a transitional XHTML DTD in a single file
- ♦ XHTML1-f-flat.dtd: a transitional XHTML DTD with frame support in a single file

In addition, as you'll learn below, it's possible to form your own DTDs that mix and match pieces of standard HTML. You can include the parts you need and leave out those you don't. You can even mix these parts with DTDs of your own devising. But

before you can do this, you'll need to take a closer look at the modules that are available for use.

The XHTML Modules

XHTML divides HTML into 28 different modules. Each module is a DTD for a particular related subset of HTML elements. Each module can be used independently of the other modules. For example, you can add basic table support to your own XML application by importing the table module into your DTD and providing definitions for a few parameter entities like `InLine` and `Flow` that include the elements of your vocabulary. The available modules include:

1. XHTML1-applet.mod
2. XHTML1-arch.mod
3. XHTML1-attrs-t.mod
4. XHTML1-attrs.mod
5. XHTML1-blkphras.mod
6. XHTML1-blkpres.mod
7. XHTML1-blkstruct.mod
8. XHTML1-charent.mod
9. XHTML1-csismap.mod
10. XHTML1-events.mod
11. XHTML1-form.mod
12. XHTML1-frames.mod
13. XHTML1-image.mod
14. XHTML1-inlphras.mod
15. XHTML1-inlpres.mod
16. XHTML1-inlstruct.mod
17. XHTML1-linking.mod
18. XHTML1-list.mod
19. XHTML1-tables.mod
20. XHTML1-meta.mod
21. XHTML1-model-t.mod
22. XHTML1-model.mod
23. XHTML1-names.mod

- 24. XHTML1-object.mod
- 25. XHTML1-script.mod
- 26. XHTML1-struct.mod
- 27. XHTML1-style.mod
- 28. XHTML1-table.mod

The frameset DTD uses all 28 modules. The transitional DTD uses most of these except the XHTML1-frames module, the XHTML1-arch module, the XHTML1-attrs module, and the XHTML1-model module. The strict DTD only uses 22, omitting the XHTML1-arch module, the XHTML1-attrs-t module, the XHTML1-frames module, the XHTML1-applet module, and the XHTML1-model-t module.

The Common Names Module

The first module all three entities import is XHTML1-names.mod, the common names module, shown in Listing 20-4.

Listing 20-4: XHTML1-names.mod: the XHTML module that defines commonly used names

```

<!-- ..... ->
<!-- XHTML 1.0 Document Common Names Module ..... ->
<!-- file: XHTML1-names.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-names.mod 1.16 99/04/01 SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ENTITIES XHTML 1.0 Common Names//EN"
SYSTEM "XHTML1-names.mod"

Revisions:
# 1999-01-31 added URIs PE for multiple URI attribute values
..... ->

<!-- i. Common Names

defines the following common names, many of these imported
from other specifications and standards.
-->

<!-- .... Imported Names .... ->

```

```
<!-- media type, as per [RFC2045] ->
<!ENTITY % ContentType "CDATA" >

<!-- comma-separated list of media types, as per [RFC2045] ->
<!ENTITY % ContentTypes "CDATA" >

<!-- a character encoding, as per [RFC2045] ->
<!ENTITY % Charset "CDATA" >

<!-- a space separated list of character encodings,
      as per [RFC2045] ->
<!ENTITY % Charsets "CDATA" >

<!-- date and time information. ISO date format ->
<!ENTITY % Datetime "CDATA" >

<!-- a single character from [ISO10646] ->
<!ENTITY % Character "CDATA" >

<!-- a language code, as per [RFC1766] ->
<!ENTITY % LanguageCode "NMTOKEN" >

<!-- space-separated list of link types ->
<!ENTITY % LinkTypes "NMTOKENS" >

<!-- single or comma-separated list of media descriptors ->
<!ENTITY % MediaDesc "CDATA" >

<!-- one or more digits (NUMBER) ->
<!ENTITY % Number "CDATA" >

<!-- a Uniform Resource Identifier, see [URI] ->
<!ENTITY % URI "CDATA" >

<!-- a space-separated list of Uniform Resource Identifiers, see
[URI] ->
<!ENTITY % URIs "CDATA" >

<!-- script expression ->
<!ENTITY % Script "CDATA" >

<!-- style sheet data ->
<!ENTITY % StyleSheet "CDATA" >

<!ENTITY % Text "CDATA" >

<!--Length defined in strict DTD for cellpadding/cellspacing-->

<!-- nn for pixels or nn% for percentage length ->
<!ENTITY % Length "CDATA" >

<!-- pixel, percentage, or relative ->
```

Continued

Listing 20-4 (continued)

```

<!ENTITY % MultiLength "CDATA" >

<!-- comma-separated list of MultiLength -->
<!ENTITY % MultiLengths "CDATA" >

<!-- integer representing length in pixels -->
<!ENTITY % Pixels "CDATA" >

<!-- render in this frame -->
<!ENTITY % FrameTarget "CDATA" >

<!-- a color using sRGB: #RRGGBB as Hex values -->
<!ENTITY % Color "CDATA" >

<!-- end of XHTML1-names.mod -->

```

DTDs aren't optimized for human legibility, even when relatively well written like this one — even less so when thrown together as is all too often the case. One of the first things you can do to understand a DTD is to reorganize it in a less formal but more legible fashion. Table 20-1 sorts the Imported Names section into a three-column table corresponding to the parameter entity name, the parameter entity value, and the comment associated with each parameter entity. This table form makes it clearer that the primary responsibility of this module is to provide parameter entities for use as element content models.

Table 20-1
Summary of Imported Names Section

| <i>Parameter Entity Name</i> | <i>Parameter Entity Value</i> | <i>Comment Associated with Parameter Entity</i> |
|------------------------------|-------------------------------|---|
| ContentType | CDATA | Media type, as per [RFC2045] |
| ContentTypes | CDATA | Comma-separated list of media types, as per [RFC2045] |
| Charset | CDATA | A character encoding, as per [RFC2045] |
| Charsets | CDATA | A space-separated list of character encodings, as per [RFC2045] |
| Datetime | CDATA | Date and time information. ISO date format |

| <i>Parameter Entity Name</i> | <i>Parameter Entity Value</i> | <i>Comment Associated with Parameter Entity</i> |
|------------------------------|-------------------------------|---|
| Character | CDATA | A single character from a single character from [ISO10646] |
| LanguageCode | CDATA | A language code, as per [RFC1766] |
| LinkTypes | NMTOKENS | Space-separated list of link types |
| MediaDesc | CDATA | Single or comma-separated list of media descriptors |
| Number | CDATA | One or more digits (NUMBER) |
| URI | CDATA | A Uniform Resource Identifier, see [URI] |
| URIs | CDATA | A space-separated list of Uniform Resource Identifiers, see [URI] |
| Script | CDATA | Script expression |
| StyleSheet | CDATA | Style sheet data |
| Text | CDATA | |
| Length | CDATA | nn for pixels or nn% for percentage length |
| MultiLength | CDATA | Pixel, percentage, or relative |
| MultiLengths | CDATA | Comma-separated list of MultiLength |
| Pixels | CDATA | Integer representing length in pixels |
| FrameTarget | CDATA | Render in this frame |
| Color | CDATA | A color using sRGB: #RRGGBB as Hex values |

What really stands out in this summary table is the number of synonyms for CDATA. In fact, all but one of these parameter entities is just a different synonym for CDATA. Why is that? It's certainly no easier to type %MultiLengths; than CDATA, even ignoring the issue of how much time it takes to remember all of these different parameter entities.

The answer is that although each of these parameter entity references resolves to simply CDATA, the use of the more descriptive parameter entity names like

`Datetime`, `FrameTarget`, or `Length` makes it more obvious to the reader of the DTD exactly what should go in a particular element or attribute value. Furthermore, the author of the DTD may look forward to a time when a schema language enables more detailed requirements to impose on attribute values. It may, at some point in the future, be possible to write declarations like this:

```
<!ATTLIST img
  src      URI          #REQUIRED
  alt      String       #REQUIRED
  longdesc URI          #IMPLIED
  height   Integer      #IMPLIED
  width    Integer      #IMPLIED
  usemap   URI          #IMPLIED
  ismap    (ismap)      #IMPLIED
  author   CDATA        #IMPLIED
  copyright CDATA       #IMPLIED
>
```

In this case, rather than having to find and replace all the places in this rather long DTD where `CDATA` is used as a length, a string, a URI, or an integer, the author can simply change the declaration of the `%Length`; `%URI`; and `%Text`; entity references like this:

```
<!ENTITY % Length "Integer">
<!ENTITY % URI    "URI">
<!ENTITY % Text   "String">
```

Almost certainly, whatever schema is eventually adopted for data-typing attributes in XML will not look exactly like the one I mocked up here. But it will likely be able to be integrated into the XHTML DTD very quickly, simply by adjusting a few of the entity declarations in the main DTD without painstakingly editing 28 modules.

The Character Entities Module

The second module all three DTDs import is XHTML1-charent.mod, shown in Listing 20-5. This module imports the DTDs that define entity sets for the standard HTML entities like `©`, ` `, and `α` for hard-to-type characters. These sets are:

- ♦ XHTML1-lat1.ent, characters 160 through 255 of Latin-1, Listing 20-30.
- ♦ XHTML1-symbol.ent, assorted useful characters and punctuation marks from outside the Latin-1 set such as the Euro sign and the em dash, Listing 20-31.
- ♦ XHTML1-special.ent, the Greek alphabet and assorted symbols commonly used for math like ∞ and \int , Listing 20-32.

Listing 20-5: XHTML1-charent.mod: the XHTML module that defines commonly used entities

```

<!-- ..... ->
<!-- XHTML 1.0 Character Entities Module
..... ->
<!-- file: XHTML1-charent.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved. Revision: @(#)XHTML1-charent.mod 1.16 99/04/01
    SMI

    This DTD module is identified by the PUBLIC and SYSTEM
    identifiers:

    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Character Entities//EN"
    SYSTEM "XHTML1-charent.mod"

    Revisions: (none)
    ..... ->

<!-- v. Character Entities for XHTML

    declares the set of character entities for XHTML,
    including Latin 1, symbol and special characters.
->

<!-- to exclude character entity declarations from a normalized
    DTD, declare %XHTML1.ents; as "IGNORE" in the internal
    subset of the dummy XHTML file used for normalization.
->
<!ENTITY % XHTML1.ents "INCLUDE" >

<![%XHTML1.ents;[
<!ENTITY % XHTML1-lat1
    PUBLIC "-//W3C//ENTITIES Latin 1//EN//XML"
    "XHTML1-lat1.ent" >
%XHTML1-lat1;

<!ENTITY % XHTML1-symbol
    PUBLIC "-//W3C//ENTITIES Symbols//EN//XML"
    "XHTML1-symbol.ent" >
%XHTML1-symbol;

<!ENTITY % XHTML1-special
    PUBLIC "-//W3C//ENTITIES Special//EN//XML"
    "XHTML1-special.ent" >
%XHTML1-special;
]]>

```

Continued

Listing 20-5 (continued)

```
<!-- end of XHTML1-charent.mod -->
```

Notice that a PUBLIC ID tries to load these entity sets. In this case, the public ID may simply be understood by a Web browser as referring to its standard HTML entity set. If not, then the relative URL giving the name of the entity set can find the necessary declarations.

The Intrinsic Events Module

The third module all three DTDs import is the intrinsic events module. This module defines the attributes for different events that can occur to different elements, and that can be scripted through JavaScript. It defines both a generic set of events that will be used for most elements (the `Events.attrib` entity) and more specific event attributes for particular elements like `form`, `button`, `label`, and `input`.

Listing 20-6: XHTML1-events.mod: the intrinsic events module

```
<!-- ..... -->
<!-- XHTML 1.0 Intrinsic Events Module ..... -->
<!-- file: XHTML1-events.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-events.mod 1.16 99/04/01
SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ENTITIES XHTML 1.0 Intrinsic Events//EN"
SYSTEM "XHTML1-events.mod"

Revisions:
#1999-01-14 transferred onfocus and onblur ATTLIST for 'a'
from link module
#1999-04-01 transferred remaining events attributes from other
modules

..... -->

<!-- iv. Intrinsic Event Attributes

These are the event attributes defined in HTML 4.0,
Section 18.2.3 "Intrinsic Events"
```

"Note: Authors of HTML documents are advised that changes are likely to occur in the realm of intrinsic events (e.g., how scripts are bound to events). Research in this realm is carried on by members of the W3C Document Object Model Working Group (see the W3C Web site at <http://www.w3.org/> for more information)."

→

```
<!ENTITY % Events.attrib
    "onclick      %Script;          #IMPLIED
     ondblclick   %Script;          #IMPLIED
     onmousedown %Script;          #IMPLIED
     onmouseup    %Script;          #IMPLIED
     onmouseover  %Script;          #IMPLIED
     onmousemove %Script;          #IMPLIED
     onmouseout   %Script;          #IMPLIED
     onkeypress   %Script;          #IMPLIED
     onkeydown    %Script;          #IMPLIED
     onkeyup      %Script;          #IMPLIED"
>
```

<!-- additional attributes on anchor element -->

```
<!ATTLIST a
    onfocus      %Script;          #IMPLIED
    onblur        %Script;          #IMPLIED
>
```

<!-- additional attributes on form element -->

```
<!ATTLIST form
    onsubmit      %Script;          #IMPLIED
    onreset       %Script;          #IMPLIED
>
```

<!-- additional attributes on label element -->

```
<!ATTLIST label
    onfocus      %Script;          #IMPLIED
    onblur        %Script;          #IMPLIED
>
```

<!-- additional attributes on input element -->

```
<!ATTLIST input
    onfocus      %Script;          #IMPLIED
    onblur        %Script;          #IMPLIED
    onselect      %Script;          #IMPLIED
    onchange      %Script;          #IMPLIED
>
```

Continued

Listing 20-6 (continued)

```
<!-- additional attributes on select element -->

<!ATTLIST select
    onfocus      %Script;          #IMPLIED
    onblur        %Script;          #IMPLIED
    onchange      %Script;          #IMPLIED
>

<!-- additional attributes on textarea element -->

<!ATTLIST textarea
    onfocus      %Script;          #IMPLIED
    onblur        %Script;          #IMPLIED
    onselect      %Script;          #IMPLIED
    onchange      %Script;          #IMPLIED
>

<!-- additional attributes on button element -->

<!ATTLIST button
    onfocus      %Script;          #IMPLIED
    onblur        %Script;          #IMPLIED
>

<!-- additional attributes on body element -->

<!ATTLIST body
    onload        %Script;          #IMPLIED
    onunload      %Script;          #IMPLIED
>

<!-- additional attributes on area element -->

<!ATTLIST area
    onfocus      %Script;          #IMPLIED
    onblur        %Script;          #IMPLIED
>

<!ENTITY % XHTML1-frames.module "IGNORE" >
<![%XHTML1-frames.module;[
<!-- additional attributes on frameset element -->

<!ATTLIST frameset
    onload        %Script;          #IMPLIED
    onunload      %Script;          #IMPLIED
>
]]>

<!-- end of XHTML1-events.mod -->
```

The values of the various attributes are all given as `%Script;`. This is a parameter entity reference that was defined back in `XHTML1-names.mod` as being equivalent to `CDATA`.

None of these elements have actually been defined yet. They will be declared in modules that are yet to be imported

The Common Attributes Modules

The next module imported declares the attributes common to most elements like `id`, `class`, `style`, and `title`. However, there are two different sets of these: one for the strict DTD and one for the transitional DTD that also provides an `align` attribute. `XHTML1-s.dtd` imports `XHTML1-attribs.mod`, shown in Listing 20-7. `XHTML1-t.dtd` imports `XHTML1-attribs-t.mod`, shown in Listing 20-8. The `.t` stands for “transitional”.

Listing 20-7: XHTML1-attribs.mod: the XHTML strict common attributes module

```
<!-- ..... -->
<!-- XHTML 1.0 Common Attributes Module ..... -->
<!-- file: XHTML1-attribs.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved. Revision: @(#)XHTML1-attribs.mod 1.14 99/04/01
    SMI

    This DTD module is identified by the PUBLIC and SYSTEM
    identifiers:

    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Common Attributes//EN"
    SYSTEM "XHTML1-attribs.mod"

    Revisions:
    # 1999-02-24 changed PE names for attribute classes to
    *.attrib;

    ..... -->

<!-- ii. Common Attributes

    This modules declares many of the common attributes for
    the Strict DTD.

-->

<!ENTITY % Core.attrib
```

Continued

Listing 20-7 (continued)

```

    "id          ID          #IMPLIED
    class       CDATA       #IMPLIED
    style       %StyleSheet; #IMPLIED
    title       %Text;      #IMPLIED"
>

<!ENTITY % I18n.attrib
    "lang       %LanguageCode; #IMPLIED
    xml:lang    %LanguageCode; #IMPLIED
    dir         (ltr|rtl)      #IMPLIED"
>

<!-- HTML intrinsic event attributes declared previously -->
<!ENTITY % Events.attrib "" >

<!ENTITY % Common.attrib
    "%Core.attrib;
    %I18n.attrib;
    %Events.attrib;" >

<!ENTITY % Align.attrib "" >

<!ENTITY % XLink.attrs "INCLUDE" >
<![%XLink.attrs;[
<!-- XLink attributes for a simple 'a' style link -->

<!ENTITY % Alink.attrib
    "xml:link    CDATA       #FIXED   'simple'
    role        CDATA       #IMPLIED
    inline      CDATA       #FIXED   'true'
    content-role CDATA       #IMPLIED
    content-title CDATA     #IMPLIED
    show        CDATA       #FIXED   'replace'
    activate    CDATA       #FIXED   'user'
    behavior    CDATA       #IMPLIED"
>
]]>
<!ENTITY % Alink.attrib "" >

<!-- end of XHTML1-attrs.mod -->

```

Listing 20-8: XHTML1-attrs-t.mod: the XHTML transitional common attributes module

```

<!-- ..... -->
<!-- XHTML 1.0 Transitional Attributes Module ..... -->
<!-- file: XHTML1-attrs-t.mod

```

```

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-attrs-t.mod 1.14 99/04/01
SMI

```

```

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

```

```

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Transitional
Attributes//EN"
SYSTEM "XHTML1-attrs-t.mod"

```

```

Revisions:
# 1999-01-24 changed PE names for attribute classes to
*.attrib;
..... ->

<!-- ii(t). Common Transitional Attributes

This modules declares the same set of common attributes as
the Strict version, but additionally includes ATTLIST
declarations for the additional attribute specifications
found in the Transitional DTD.
->

<!ENTITY % Core.attrib
    "id          ID          #IMPLIED
     class       CDATA       #IMPLIED
     style       %StyleSheet; #IMPLIED
     title       %Text;      #IMPLIED"
>

<!ENTITY % I18n.attrib
    "lang        %LanguageCode; #IMPLIED
     xml:lang    %LanguageCode; #IMPLIED
     dir         (ltr|rtl)     #IMPLIED"
>

<!-- HTML intrinsic event attributes declared previously ->

<!ENTITY % Common.attrib
    "%Core.attrib;
     %I18n.attrib;
     %Events.attrib;"
>

<!-- horizontal text alignment ->
<!ENTITY % Align.attrib
    "align       (left|center|right|justify) #IMPLIED"
>

```

Continued

Listing 20-8 (continued)

```

<!-- horizontal and vertical alignment -->
<!ENTITY % IAlign.attrib
    "align (top|middle|bottom|left|right) #IMPLIED"
>

<!ENTITY % XLink.attrs "INCLUDE" >
<![%XLink.attrs;[
<!-- XLink attributes for a simple anchor link -->

<!ENTITY % Alink.attrib
    "xml:link          CDATA          #FIXED   'simple'
     role              CDATA          #IMPLIED
     inline            CDATA          #FIXED   'true'
     content-role      CDATA          #IMPLIED
     content-title     CDATA          #IMPLIED
     show              CDATA          #FIXED   'replace'
     activate          CDATA          #FIXED   'user'
     behavior          CDATA          #IMPLIED"
>
]]>
<!ENTITY % Alink.attrib "" >

<!-- end of XHTML1-attrs-t.mod -->

```

Aside from the `align` attributes (which are only included by the transitional DTD), these two modules are very similar. They define parameter entities for attributes (and groups of attributes) that can apply to any (or almost any) HTML element. These parameter entities are used inside `ATTLIST` declarations in other modules.

To grasp this section, let's use a different trick. Pretend we're cheating on one of those fast food restaurant menu mazes, and work backwards from the goal rather than forwards from the start. Consider the `Common.attrib` entity:

```

<!ENTITY % Common.attrib
    "%Core.attrib;
     %I18n.attrib;
     %Events.attrib;"
>

```

This entity sums up those attributes that apply to almost any element and will serve as the first part of most `ATTLIST` declarations in the individual modules. For example:

```

<!ATTLIST address
    %Common.attrib;
>

```

The last item in the declaration of `Common.attrib` is `%Events.attrib`; This is defined as an empty string in `XHTML1-attribs.mod`.

```
<!-- HTML intrinsic event attributes declared previously -->
<!ENTITY % Events.attrib "" >
```

However, as the comment indicates, this can be overridden in the base DTD to add attributes to the ones normally present. In particular, it was overridden in Listing 20-6 like this:

```
<!ENTITY % Events.attrib
"onclick      %Script;          #IMPLIED
 ondblclick   %Script;          #IMPLIED
 onmousedown  %Script;          #IMPLIED
 onmouseup    %Script;          #IMPLIED
 onmouseover  %Script;          #IMPLIED
 onmousemove  %Script;          #IMPLIED
 onmouseout   %Script;          #IMPLIED
 onkeypress   %Script;          #IMPLIED
 onkeydown    %Script;          #IMPLIED
 onkeyup      %Script;          #IMPLIED"
>
```

The `%Script`; parameter entity reference was defined in Listing 20-4, `XHTML1-names.mod` as `CDATA`. Thus the replacement text of `Common.attrib` looks like this:

```
%Core.attrib;
%I18n.attrib;
onclick      CDATA          #IMPLIED
 ondblclick   CDATA          #IMPLIED
 onmousedown  CDATA          #IMPLIED
 onmouseup    CDATA          #IMPLIED
 onmouseover  CDATA          #IMPLIED
 onmousemove  CDATA          #IMPLIED
 onmouseout   CDATA          #IMPLIED
 onkeypress   CDATA          #IMPLIED
 onkeydown    CDATA          #IMPLIED
 onkeyup      CDATA          #IMPLIED
```

The second to last item in the declaration of `Common.attrib` is `%I18n.attrib`; This is defined in the same module with this declaration:

```
<!ENTITY % I18n.attrib
"lang         %LanguageCode;    #IMPLIED
 xml:lang     %LanguageCode;    #IMPLIED
 dir          (ltr|rtl)         #IMPLIED"
>
```

The `%LanguageCode;` parameter entity reference was also defined in `XHTML1-names.mod` as an alias for `CDATA`. Including these, `%Common.attrib;` now expands to:

```
%Core.attrib;
lang          CDATA; #IMPLIED
xml:lang      CDATA      #IMPLIED
dir           (ltr|rtl)  #IMPLIED
onclick       CDATA      #IMPLIED
ondblclick    CDATA      #IMPLIED
onmousedown   CDATA      #IMPLIED
onmouseup     CDATA      #IMPLIED
onmouseover   CDATA      #IMPLIED
onmousemove   CDATA      #IMPLIED
onmouseout    CDATA      #IMPLIED
onkeypress    CDATA      #IMPLIED
onkeydown     CDATA      #IMPLIED
onkeyup       CDATA      #IMPLIED
```

The last remaining parameter entity reference to expand is `%Core.attrib;`. This is also declared in `XHTML1-attribs.mod` as:

```
<!ENTITY % Core.attrib
      "id          ID          #IMPLIED
      class       CDATA       #IMPLIED
      style       %StyleSheet; #IMPLIED
      title       %Text;      #IMPLIED"
>
```

This declaration includes two more parameter entity references: `%StyleSheet;` and `%Text;`. Each of these expands to `CDATA`., again from previous declarations in `XHTML1-names.mod`. Thus, the final expansion of `%Common.attrib;` is:

```
id          ID          #IMPLIED
class       CDATA       #IMPLIED
style       CDATA       #IMPLIED
title       CDATA       #IMPLIED
lang        CDATA       #IMPLIED
xml:lang    CDATA       #IMPLIED
dir         (ltr|rtl)  #IMPLIED
onclick     CDATA       #IMPLIED
ondblclick  CDATA       #IMPLIED
onmousedown CDATA       #IMPLIED
onmouseup   CDATA       #IMPLIED
onmouseover CDATA       #IMPLIED
onmousemove CDATA       #IMPLIED
onmouseout  CDATA       #IMPLIED
onkeypress  CDATA       #IMPLIED
onkeydown   CDATA       #IMPLIED
onkeyup     CDATA       #IMPLIED
```

Note

I've been a little cavalier with whitespace in this example. The true expansion of `%Common.attrib;` isn't so nicely formatted. However, whitespace is insignificant in declarations so this isn't really important, and you should feel free to manually adjust whitespace to line columns up or insert line breaks when manually expanding a parameter entity reference to see what it says.

Thus, `%Common.attrib;` has subsumed most of the other material in this section. You won't see `%Core.attrib;` or `%I18N.attrib;` or `%Events.attrib;` often again in later modules. They're just like private methods in C++ that could be inlined but aren't solely for the sake of efficiency.

The XLink attributes are not subsumed into `%Common.attrib;`. That's because although many elements can possess the link attributes, many cannot. Thus, when the XLink attributes are added to an element, you must use a separate parameter entity reference, `%Alink.attrib;`.

The Document Model Module

The XHTML DTDs next import a module that declares entities for all the text flow elements like `p`, `div`, and `blockquote`. These are the elements that form the basic tree structure of a well-formed HTML document. Again, two separate modules are provided; one for the strict DTD (Listing 20-9, `XHTML1-model.mod`) and one for the transitional DTD (Listing 20-10, `XHTML1-model-t.mod`).

Listing 20-9: XHTML1-model.mod: the strict document model module

```
<!-- ..... ->
<!-- XHTML 1.0 Document Model Module ..... ->
<!-- file: XHTML1-model.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-model.mod 1.12 99/04/01 SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Document Model//EN"
SYSTEM "XHTML1-model.mod"

Revisions:
(none)
..... ->

<!-- iii. Document Model
```

Continued

Listing 20-9 (continued)

This module declares entities describing all text flow elements, excluding Transitional elements. This module describes the groupings of elements that make up HTML's document model.

HTML has two basic content models:

```

    %Inline.mix;  character-level elements
    %Block.mix;  block-like elements, eg., paragraphs and
lists
```

The reserved word '#PCDATA' (indicating a text string) is now included explicitly with each element declaration, as XML requires that the reserved word occur first in a content model specification..

→

```
<!-- ..... Miscellaneous Elements ..... -->
```

```
<!-- These elements are neither block nor inline, and can
essentially be used anywhere in the document body -->
```

```
<!ENTITY % Misc.class
    "ins | del | script | noscript" >
```

```
<!-- ..... Inline Elements ..... -->
```

```
<!ENTITY % Inlstruct.class
    "bdo | br | span" >
```

```
<!ENTITY % Inlpres.class "tt | i | b | big | small | sub |
sup" >
```

```
<!ENTITY % Inlphras.class
    "em | strong | dfn | code | samp | kbd | var | cite | abbr
| acronym | q" >
```

```
<!ENTITY % Inlspecial.class "a | img | object | map" >
```

```
<!ENTITY % Formctrl.class "input | select | textarea | label |
button" >
```

```
<!-- %Inline.class; includes all inline elements, used as a
component in mixes -->
```

```
<!ENTITY % Inline.class
    "%Inlstruct.class;
    | %Inlpres.class;
    | %Inlphras.class;
    | %Inlspecial.class;
```

```

    | %Formctrl.class;"
>

<!-- %Inline.mix; includes all inline elements, including
%Misc.class; -->

<!ENTITY % Inline.mix
    "%Inline.class;
    | %Misc.class;"
>

<!-- %Inline-noa.class; includes all non-anchor inlines,
    used as a component in mixes -->

<!ENTITY % Inline-noa.class
    "%Inlstruct.class;
    | %Inlpres.class;
    | %Inlphras.class;
    | img | object | map
    | %Formctrl.class;"
>

<!-- %Inline-noa.mix; includes all non-anchor inlines -->

<!ENTITY % Inline-noa.mix
    "%Inline-noa.class;
    | %Misc.class;"
>

<!-- ..... Block Elements ..... -->

<!-- In the HTML 4.0 DTD, heading and list elements were
    included in the %block; parameter entity. The
    %Heading.class; and %List.class; parameter entities must
    now be included explicitly on element declarations where
    desired.
-->

<!-- There are six levels of headings from H1 (the most
    important) to H6 (the least important).
-->
<!ENTITY % Heading.class "h1 | h2 | h3 | h4 | h5 | h6" >
<!ENTITY % List.class "ul | ol | dl" >
<!ENTITY % Blkstruct.class "p | div" >
<!ENTITY % Blkpres.class "hr" >
<!ENTITY % Blkphras.class "pre | blockquote | address" >
<!ENTITY % Blkform.class "form | fieldset" >

```

Continued

Listing 20-9 (continued)

```

<!ENTITY % Blkspecial.class "table" >

<!-- %Block.class; includes all block elements,
      used as an component in mixes -->

<!ENTITY % Block.class
      "%Blkstruct.class;
      | %Blkpres.class;
      | %Blkphras.class;
      | %Blkform.class;
      | %Blkspecial.class;"
>

<!-- %Block.mix; includes all block elements plus %Misc.class;
-->

<!ENTITY % Block.mix
      "%Block.class;
      | %Misc.class;"
>

<!-- %Block-noform.class; includes all non-form block elements,
      used as a component in mixes -->

<!ENTITY % Block-noform.class
      "%Blkstruct.class;
      | %Blkpres.class;
      | %Blkphras.class;
      | %Blkspecial.class;"
>

<!-- %Block-noform.mix; includes all non-form block elements,
      plus %Misc.class; -->

<!ENTITY % Block-noform.mix
      "%Block-noform.class;
      | %Misc.class;"
>

<!-- ..... All Content Elements ..... -->

<!-- %Flow.mix; includes all text content, block and inline -->

<!ENTITY % Flow.mix
      "%Heading.class;
      | %List.class;
      | %Block.class;"

```

```

    | %Inline.class;
    | %Misc.class;"
>
<!-- end of XHTML1-model.mod -->

```

Listing 20-10: XHTML1-model-t.mod: the transitional document model module

```

<!-- ..... ->
<!-- XHTML 1.0 Transitional Text Markup Module ..... ->
<!-- file: XHTML1-model-t.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-model-t.mod 1.14 99/04/01
SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Transitional Document
Model//EN" SYSTEM "XHTML1-model-t.mod"

Revisions:
#1999-01-14 rearranged forms and frames PEs, adding
%Blkform.class;
..... ->

<!-- iii(t). Transitional Document Model

This modules declares entities describing all text flow
elements, including Transitional elements. This module
describes the groupings of elements that make up HTML's
document model.

HTML has two basic content models:

    %Inline.mix; character-level elements
    %Block.mix; block-like elements, eg., paragraphs and
lists

The reserved word '#PCDATA' (indicating a text string) is
now included explicitly with each element declaration, as
XML requires that the reserved word occur first in a

```

Continued

Listing 20-10 (continued)

```

        content model specification..
->

<!-- ..... Miscellaneous Elements .....
->

<!-- These elements are neither block nor inline, and can
        essentially be used anywhere in the document body ->

<!ENTITY % Misc.class
        "ins | del | script | noscript" >

<!-- ..... Inline Elements ..... ->

<!ENTITY % Inlstruct.class
        "bdo | br | span" >

<!ENTITY % Inlpres.class
        "tt | i | b | u | s | strike | big | small | font | basefont
         | sub | sup" >

<!ENTITY % Inlphras.class
        "em | strong | dfn | code | samp | kbd | var | cite
         | abbr | acronym | q" >

<![%XHTML1-frames.module;[
<!-- %Inlspecial.class; includes iframe in Frameset DTD version
->

<!ENTITY % Inlspecial.class "a | img | applet | object | map
                             | iframe">
]]>

<!ENTITY % Inlspecial.class "a | img | applet | object | map">

<!ENTITY % Formctrl.class "input | select | textarea | label
                           | button">

<!-- %Inline.class; includes all inline elements, used
        as a component in mixes ->

<!ENTITY % Inline.class
        "%Inlstruct.class;
         | %Inlpres.class;
         | %Inlphras.class;
         | %Inlspecial.class;
         | %Formctrl.class;"
>

<!-- %Inline.mix; includes all inline elements,
        including %Misc.class; ->

```

```

<!ENTITY % Inline.mix
    "%Inline.class;
     | %Misc.class;"
>

<!-- %Inline-noa.class; includes all non-anchor inlines,
     used as a component in mixes -->

<!ENTITY % Inline-noa.class
    "%Inlstruct.class;
     | %Inlpres.class;
     | %Inlphras.class;
     | img | applet | object | map
     | %Formctrl.class;"
>

<!-- %Inline-noa.mix; includes all non-anchor inlines -->

<!ENTITY % Inline-noa.mix
    "%Inline-noa.class;
     | %Misc.class;"
>

<!-- ..... Block Elements ..... -->

<!-- In the HTML 4.0 DTD, heading and list elements were
     included in the %block; parameter entity. The
     %Heading.class; and %List.class; parameter entities must
     now be included explicitly on element declarations where
     desired.
-->

<!-- There are six levels of headings from h1 (the most
     important)
     to h6 (the least important).
-->
<!ENTITY % Heading.class "h1 | h2 | h3 | h4 | h5 | h6">
<!ENTITY % List.class "ul | ol | dl | menu | dir" >
<!ENTITY % Blkstruct.class "p | div" >
<!ENTITY % Blkpres.class "center | hr" >
<!ENTITY % Blkphras.class "pre | blockquote | address" >
<!ENTITY % Blkform.class "form | fieldset" >

<![%XHTML1-frames.module;[
<!-- Blkspecial.class includes noframes in Frameset DTD version
-->

```

Continued

Listing 20-10 (continued)

```

<!ENTITY % Blkspecial.class "noframes | table" >
]]>

<!ENTITY % Blkspecial.class "table" >

<!-- %Block.class; includes all block elements,
      used as an component in mixes ->

<!ENTITY % Block.class
      "%Blkstruct.class;
      | %Blkpres.class;
      | %Blkphras.class;
      | %Blkform.class;
      | %Blkspecial.class;"
>

<!-- %Block.mix; includes all block elements plus %Misc.class; -
>

<!ENTITY % Block.mix
      "%Block.class;
      | %Misc.class;"
>

<!-- %Block-noform.class; includes all non-form block elements,
      used as a component in mixes ->

<!ENTITY % Block-noform.class
      "%Blkstruct.class;
      | %Blkpres.class;
      | %Blkphras.class;
      | %Blkspecial.class;"
>

<!-- %Block-noform.mix; includes all non-form block elements,
      plus %Misc.class; ->

<!ENTITY % Block-noform.mix
      "%Block-noform.class;
      | %Misc.class;"
>

<!-- ..... All Content Elements ..... ->

<!-- %Flow.mix; includes all text content, block and inline ->

<!ENTITY % Flow.mix
      "%Heading.class;
      | %List.class;
      | %Block.class;

```

```

    | %Inline.class;
    | %Misc.class;"
>

<!-- end of XHTML1-model-t.mod -->

```

The elements themselves are not what's declared in these two modules, but rather entities that can be used in content models for these elements and the elements that contain them. The actual element declarations come later.

These modules are divided into logical sections denoted by comments. The first is the Miscellaneous Elements section. This defines the `Misc.class` parameter entity for four elements that may appear as either inline or block elements:

```

<!ENTITY % Misc.class
    "ins | del | script | noscript" >

```

Next, the Inline Elements section defines the inline elements of HTML, those elements that may not contain block level elements. Here the transitional and strict DTDs differ in exactly which elements they include. However, they both divide the inline elements into structural (`Inlstruct.class`), presentational (`Inlpres.class`), phrasal (`Inlphras.class`), special (`Inlspecial.class`), and form (`Formctrl.class`) classes. These intermediate parameter entities are combined to form the `Inline.class` parameter entity which lists all the elements that may appear as inline elements. Then `%Inline.class;` is combined with the previously defined `%Misc.class;` parameter entity reference to create the `Inline.mix` parameter entity that includes both inline and miscellaneous elements.

```

<!ENTITY % Inline.mix
    "%Inline.class;
    | %Misc.class;"
>

```

A similar parameter entity called `Inline-noa.class` is also defined. Here `noa` stands for “no a element”. This one element is left out because it will be needed elsewhere when the block-level entities are defined next. Including it here has the potential to lead to ambiguous content models; not a major disaster but something to be avoided if possible.

The Block Elements section lists the different kinds of block-level elements, and defines parameter entities for each. This builds up in steps to the final `%Block.class;` parameter entity reference, which lists all block-level elements and `%Flow.mix;` which lists all block and inline elements.

Parameter entities are defined for headings `h1` through `h6` (`Heading.class`) and lists (`List.class`). Block-level parameter entities include structural blocks `p` and `div` (`Blkstruct.class`), presentational blocks, particularly `hr`, (`Blkpres.class`), forms and fieldsets (`Blkform.class`), and tables (`Blkspecial.class`). These are all combined in the `Block.class` parameter entity. This is merged with the `Misc.class` parameter entity to form the `Block.mix` parameter entity that contains both block-level and miscellaneous elements. Finally, `Block-noform.class` and a `Block-noform.mix` entities are defined to be used when all block-level elements, except forms, are desired.

The final Content Elements section defines `Flow.mix`, which pulls together all of the above: block, inline, heading, list, and miscellaneous.

```
<!ENTITY % Flow.mix
        "%Heading.class;
        |%List.class;
        |%Block.class;
        |%Inline.class;
        |%Misc.class;"
>
```

The Inline Structural Module

The next module, `XHTML1-inlstruct.mod`, shown in Listing 20-11, is used by both the transitional and the strict DTDs to define the inline structural elements `bdo`, `br`, `del`, `ins`, and `span`.

Listing 20-11: XHTML1-inlstruct.mod: the inline structural module

```
<!-- ..... -->
<!-- XHTML 1.0 Inline Phrasal Module ..... -->
<!-- file: XHTML1-inlstruct.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-inlstruct.mod 1.10 99/04/01
SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Structural//EN"
SYSTEM "XHTML1-inlstruct.mod"

Revisions:
(none)
..... -->
```

```

<!-- c1. Inline Structural
      bdo, br, del, ins, span
->

<!ENTITY % Bdo.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT bdo %Bdo.content; >
<!ATTLIST bdo
      %Core.attrib;
      lang      %LanguageCode;          #IMPLIED
      dir      (ltr|rtl)                #REQUIRED
>

<!ENTITY % Br.content "EMPTY" >
<!ELEMENT br %Br.content; >
<!ATTLIST br
      %Core.attrib;
>

<!ENTITY % Del.content "( #PCDATA | %Flow.mix; )*" >
<!ELEMENT del %Del.content; >
<!ATTLIST del
      %Common.attrib;
      cite      %URI;                    #IMPLIED
      datetime  %Datetime;              #IMPLIED
>

<!ENTITY % Ins.content "( #PCDATA | %Flow.mix; )*" >
<!ELEMENT ins %Ins.content; >
<!ATTLIST ins
      %Common.attrib;
      cite      %URI;                    #IMPLIED
      datetime  %Datetime;              #IMPLIED
>

<!ENTITY % Span.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT span %Span.content; >
<!ATTLIST span
      %Common.attrib;
>

<!-- end of XHTML1-inlstruct.mod ->

```

This module actually begins to use the parameter entities the last several modules have defined. In particular, it defines the attributes of `del`, `ins`, and `span` as `%Common.attrib`; and those of `bdo` and `br` as `%Core.attrib`. It also uses several

of the CDATA aliases from XHTML1-names.mod; specifically, %LanguageCode;, %URI; and %Datetime;.

Also note that the content models for elements are given as locally declared entities. For example:

```
<!ENTITY % Span.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT span %Span.content; >
```

Why not simply declare them without the extra parameter entity reference like the following?

```
<!ELEMENT span ( #PCDATA | %Inline.mix; )* >
```

The reason is simple: using the parameter entity reference allows other modules to override this content model. These aren't necessarily the modules used here, but modules from completely different XML applications that may be merged with the XHTML modules.

Inline Presentational Module

The next module, XHTML1-inlpres.mod, shown in Listing 20-12, is used by both the transitional and the strict DTDs to define the inline presentational elements b, big, i, small, sub, sup, and tt.

Listing 20-12: XHTML1-inlpres.mod: the inline presentational module

```
<!-- ..... -->
<!-- XHTML 1.0 Inline Presentational Module ..... -->
<!-- file: XHTML1-inlpres.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-inlpres.mod 1.13 99/04/01
SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline
Presentational//EN" SYSTEM "XHTML1-inlpres.mod"

Revisions:
(none)
..... -->

<!-- c3. Inline Presentational
```

b, big, i, small, sub, sup, tt

A conditional section includes additional declarations for the Transitional DTD

basefont, font, s, strike, u

→

```

<!ENTITY % B.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT b %B.content; >
<!ATTLIST b
    %Common.attrib;
>

<!ENTITY % Big.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT big %Big.content; >
<!ATTLIST big
    %Common.attrib;
>

<!ENTITY % I.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT i %I.content; >
<!ATTLIST i
    %Common.attrib;
>

<!ENTITY % Small.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT small %Small.content; >
<!ATTLIST small
    %Common.attrib;
>

<!ENTITY % Sub.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT sub %Sub.content; >
<!ATTLIST sub
    %Common.attrib;
>

<!ENTITY % Sup.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT sup %Sup.content; >
<!ATTLIST sup
    %Common.attrib;
>

<!ENTITY % Tt.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT tt %Tt.content; >
<!ATTLIST tt
    %Common.attrib;
>

<![%XHTML.Transitional;[

```

Continued

Listing 20-12 (continued)

```

<!ENTITY % Basefont.content "EMPTY" >
<!ELEMENT basefont %Basefont.content; >
<!ATTLIST basefont
    id          ID                #IMPLIED
    size        CDATA             #REQUIRED
    color       %Color;           #IMPLIED
    face        CDATA             #IMPLIED
>

<!ENTITY % Font.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT font %Font.content; >
<!ATTLIST font
    %Core.attrib;
    %I18n.attrib;
    size        CDATA             #IMPLIED
    color       %Color;           #IMPLIED
    face        CDATA             #IMPLIED
>

<!ENTITY % S.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT s %S.content; >
<!ATTLIST s
    %Common.attrib;
>

<!ENTITY % Strike.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT strike %Strike.content; >
<!ATTLIST strike
    %Common.attrib;
>

<!ENTITY % U.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT u %U.content; >
<!ATTLIST u
    %Common.attrib;
>

]]>

<!-- end of XHTML1-inlpres.mod -->

```

There's a neat trick in this file that defines the deprecated `basefont`, `font`, `s`, `strike`, and `u` elements for the transitional DTD but not for the strict DTD. The

declarations for these elements and their attributes are all wrapped in this construct:

```
<![%XHTML.Transitional;[
  <!-- basefont, font, s, strike, and u declarations -->
]]>
```

Recall that `XHTML-t.dtd` defined the parameter entity `XHTML.Transitional` as `INCLUDE` but the `XHTML-s.dtd` defined it as `IGNORE`. Thus these declarations are included by the transitional DTD and ignored by the strict one.

Inline Phrasal Module

The next module, `XHTML1-inlphras.mod`, shown in Listing 20-13, is used by both the transitional and the strict DTDs to define the inline phrasal elements: `abbr`, `acronym`, `cite`, `code`, `dfn`, `em`, `kbd`, `q`, `samp`, `strong`, and `var`.

Listing 20-13: XHTML1-inlphras.mod: the inline phrasal module

```
<!-- ..... -->
<!-- XHTML 1.0 Inline Phrasal Module ..... -->
<!-- file: XHTML1-inlphras.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-inlphras.mod 1.14 99/04/01
SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Phrasal//EN"
SYSTEM "XHTML1-inlphras.mod"

Revisions:
#1999-01-29 moved bdo, br, del, ins, span to inline
structural module
..... -->

<!-- c2. Inline Phrasal

      abbr, acronym, cite, code, dfn, em, kbd, q, samp,
strong, var
-->

<!ENTITY % Abbr.content "( #PCDATA | %Inline.mix; )*" >
```

Continued

Listing 20-13 (continued)

```

<!ELEMENT abbr %Abbr.content; >
<!ATTLIST abbr
    %Common.attrib;
>

<!ENTITY % Acronym.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT acronym %Acronym.content; >
<!ATTLIST acronym
    %Common.attrib;
>

<!ENTITY % Cite.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT cite %Cite.content; >
<!ATTLIST cite
    %Common.attrib;
>

<!ENTITY % Code.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT code %Code.content; >
<!ATTLIST code
    %Common.attrib;
>

<!ENTITY % Dfn.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT dfn %Dfn.content; >
<!ATTLIST dfn
    %Common.attrib;
>

<!ENTITY % Em.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT em %Em.content; >
<!ATTLIST em
    %Common.attrib;
>

<!ENTITY % Kbd.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT kbd %Kbd.content; >
<!ATTLIST kbd
    %Common.attrib;
>

<!ENTITY % Q.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT q %Q.content; >
<!ATTLIST q
    %Common.attrib;
    cite %URI; #IMPLIED
>

<!ENTITY % Samp.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT samp %Samp.content; >
<!ATTLIST samp

```

```

    %Common.attrib;
>
<!ENTITY % Strong.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT strong %Strong.content; >
<!ATTLIST strong
    %Common.attrib;
>

<!ENTITY % Var.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT var %Var.content; >
<!ATTLIST var
    %Common.attrib;
>

<!-- end of XHTML1-inlphras.mod -->

```

With the exception of `q`, all these inline elements in this module have identical content models and identical attribute lists. They may all contain `#PCDATA | %Inline.mix`; and they all have `%Common.attrib` attributes. The `q` element can have all of these, too. However, it may also have one additional optional attribute, `cite`, which should contain a URI pointing to the source of the quotation.

This example demonstrates the power of the parameter entity approach particularly well. Without parameter entity references, this module would appear several times longer and several times less easy to grasp as a whole.

Block Structural Module

The next module, `XHTML1-blkstruct.mod`, shown in Listing 20-14, is a very simple module used by both the transitional and the strict DTDs to define the `p` and the `div` block-level structural elements.

Listing 20-14: XHTML1-blkstruct.mod: the inline phrasal module

```

<!-- ..... ->
<!-- XHTML 1.0 Block Structural Module ..... ->
<!-- file: XHTML1-blkstruct.mod

```

```

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved. Revision: @(#)XHTML1-blkstruct.mod 1.10 99/04/01
SMI

```

Continued

Listing 20-14 (continued)

```

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Structural//EN"
SYSTEM "XHTML1-blkstruct.mod"

Revisions:
(none)
..... ->

<!-- b1. Block Structural

      div, p
->

<!ENTITY % Div.content "( #PCDATA | %Flow.mix; )*" >
<!ELEMENT div %Div.content; >
<!ATTLIST div
      %Common.attrib;
      %Align.attrib;
>

<!ENTITY % P.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT p %P.content; >
<!ATTLIST p
      %Common.attrib;
>

<!-- end of XHTML1-blkstruct.mod ->

```

Block-Presentational Module

The next module, `XHTML1-blkpres.mod`, shown in Listing 20-15, defines the `hr` and the `center` block-level structural elements for both the transitional and the strict DTDs.

Listing 20-15: XHTML1-blkpres.mod: the inline presentational module

```

<!-- ..... ->
<!-- XHTML 1.0 Block Presentational Module ..... ->
<!-- file: XHTML1-blkpres.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights

```

```

Reserved. Revision: @(#)XHTML1-blkpres.mod 1.15 99/04/01
SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block
Presentational//EN" SYSTEM "XHTML1-blkpres.mod"

Revisions:
# 1999-01-31  added I18n.attrib to hr (errata)
..... ->

<!-- b3. Block Presentational

      hr

      A conditional section includes additional declarations
      for the Transitional DTD

      center
->

<!ENTITY % Hr.content  "EMPTY" >
<!ELEMENT hr  %Hr.content; >
<!ATTLIST hr
      %Core.attrib;
      %I18n.attrib;
      %Events.attrib;
>

<![%XHTML.Transitional;[
<!ENTITY % Center.content "( #PCDATA | %Flow.mix; )*" >
<!ELEMENT center %Center.content; >
<!ATTLIST center
      %Common.attrib;
>

<!-- additional attributes on hr ->
<!ATTLIST hr
      align      (left|center|right)      #IMPLIED
      noshade    (noshade)                #IMPLIED
      size       %Pixels;                  #IMPLIED
      width      %Length;                  #IMPLIED
>
]]>

<!-- end of XHTML1-blkpres.mod ->

```

The `center` element is deprecated in HTML 4.0 so it's placed in the `<![%XHTML.Transitional;[]]>` region that will be included by the transitional DTD and ignored by the strict DTD. The `hr` element is included by both. However, some (but not all) of its attributes are deprecated in HTML 4.0. Consequently, it has two ATTLIST declarations, one for the undeprecated attributes and one for the deprecated attributes. The ATTLIST for the deprecated attributes is placed in the `<![%XHTML.Transitional;[]]>` region so it will be ignored by the strict DTD.

Block-Phrasal Module

The next module, `XHTML1-blkphras.mod`, shown in Listing 20-16, is a very simple module used by both the transitional and the strict DTDs to define the `address`, `blockquote`, `pre`, `h1`, `h2`, `h3`, `h4`, `h5`, and `h6` block-level phrasal elements.

Listing 20-16: XHTML1-blkphras.mod: the block-phrasal module

```
<!-- ..... ->
<!-- XHTML 1.0 Block Phrasal Module ..... ->
<!-- file: XHTML1-blkphras.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved. Revision: @(#)XHTML1-blkphras.mod 1.13 99/04/01
    SMI

    This DTD module is identified by the PUBLIC and SYSTEM
    identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Phrasal//EN"
    SYSTEM "XHTML1-blkphras.mod"

    Revisions:
    # 1998-11-10 removed pre exclusions - content model
                  changed to mimic HTML 4.0
    # 1999-01-29 moved div and p to block structural module
                  ..... ->

<!-- b2. Block Phrasal

        address, blockquote, pre, h1, h2, h3, h4, h5, h6
->

<!ENTITY % Address.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT address %Address.content; >
<!ATTLIST address
        %Common.attrib;
>
```

```

<![%XHTML.Transitional;[
<!ENTITY % Blockquote.content "( %Flow.mix; )*" >
]]>
<!ENTITY % Blockquote.content
      "( %Heading.class;
       | %List.class;
       | %Block.mix; )+"
>

<!ELEMENT blockquote %Blockquote.content; >
<!ATTLIST blockquote
      %Common.attrib;
      cite          %URI;                  #IMPLIED
>

<!ENTITY % Pre.content
      "( #PCDATA | tt | i | b
       | %Inlstruct.class; | %Inlphras.class;
       | a | script | map
       | %Formctrl.class; )*"
>

<!ELEMENT pre %Pre.content; >
<!ATTLIST pre
      %Common.attrib;
      xml:space     CDATA                  #FIXED "preserve"
>

<!-- ..... Heading Elements ..... ->

<!ENTITY % Heading.content "( #PCDATA | %Inline.mix; )*" >

<!ELEMENT h1 %Heading.content; >
<!ATTLIST h1
      %Common.attrib;
      %Align.attrib;
>

<!ELEMENT h2 %Heading.content; >
<!ATTLIST h2
      %Common.attrib;
      %Align.attrib;
>

<!ELEMENT h3 %Heading.content; >
<!ATTLIST h3
      %Common.attrib;
      %Align.attrib;
>

<!ELEMENT h4 %Heading.content; >
<!ATTLIST h4

```

Continued

Listing 20-16 (continued)

```

        %Common.attrib;
        %Align.attrib;
    >

<!ELEMENT h5 %Heading.content; >
<!ATTLIST h5
        %Common.attrib;
        %Align.attrib;
>

<!ELEMENT h6 %Heading.content; >
<!ATTLIST h6
        %Common.attrib;
        %Align.attrib;
>

<!-- end of XHTML1-blkphras.mod -->

```

Once again, the `<![%XHTML.Transitional;[]]>` region separates the declarations for the strict DTD from those for the transitional DTD. Here it's the content model of the `blockquote` element that's adjusted depending on which DTD is being used in these lines:

```

<![%XHTML.Transitional;[
<!ENTITY % Blockquote.content "( %Flow.mix; )*" >
]]>
<!ENTITY % Blockquote.content
        "( %Heading.class;
         | %List.class;
         | %Block.mix; )+"
>

```

The first definition of `Blockquote.content` is used only with the transitional DTD. If it is included, it takes precedence over the second redefinition. However, with the strict DTD, only the second definition is ever seen or used.

The Scripting Module

The next module, `XHTML1-script.mod`, shown in Listing 20-17, is a very simple module used by both the transitional and the strict DTDs to define the `script` and `noscript` elements.

Listing 20-17: XHTML1-script.mod: the scripting module

```

<!-- ..... ->
<!-- XHTML 1.0 Document Scripting Module ..... ->
<!-- file: XHTML1-script.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved.
    Revision: @(#)XHTML1-script.mod 1.13 99/04/01 SMI

    This DTD module is identified by the PUBLIC
    and SYSTEM identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Scripting//EN"
    SYSTEM "XHTML1-script.mod"

    Revisions:
    # 1999-01-30 added xml:space to script
    # 1999-02-01 removed for and event attributes from script
    ..... ->

<!-- d4. Scripting

    script, noscript
->

<!ENTITY % Script.content "( #PCDATA )" >
<!ELEMENT script %Script.content; >
<!ATTLIST script
    charset      %Charset;           #IMPLIED
    type         %ContentType;       #REQUIRED
    src          %URI;               #IMPLIED
    defer        (defer)             #IMPLIED
    xml:space    CDATA               #FIXED 'preserve'
>

<!ENTITY % Noscript.content
    "( %Heading.class;
    | %List.class;
    | %Block.mix; )+"
>
<!ELEMENT noscript %Noscript.content; >
<!ATTLIST noscript
    %Common.attrib;
>

<!-- end of XHTML1-script.mod ->

```

The Stylesheets Module

The next module, XHTML1-style.mod, shown in Listing 20-18, is a particularly simple module used by both the transitional and the strict DTDs to define a single element, `style`.

Listing 20-18: XHTML1-style.mod: the stylesheets module

```

<!-- ..... -->
<!-- XHTML 1.0 Document Stylesheet Module ..... -->
<!-- file: XHTML1-style.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved.
    Revision: @(#)XHTML1-style.mod 1.1399/04/01 SMI

    This DTD module is identified by the PUBLIC and SYSTEM
    identifiers:

    PUBLIC "-//W3C//DTD XHTML 1.0 Stylesheets//EN"
    SYSTEM "XHTML1-style.mod"

    Revisions:
    # 1999-01-30 added xml:space to style
    ..... -->

<!-- d5. Stylesheets

    style
-->

<!ENTITY % Style.content "( #PCDATA )" >
<!ELEMENT style %Style.content; >
<!ATTLIST style
    %I18n.attrib;
    type           %ContentType;           #REQUIRED
    media          %MediaDesc;             #IMPLIED
    title          %Text;                  #IMPLIED
    xml:space      CDATA                    #FIXED 'preserve'
>

<!-- end of XHTML1-style.mod -->

```

The Image Module

The next module, `XHTML1-image.mod`, shown in Listing 20-19, is another particularly simple module used by both the transitional and the strict DTDs to define a single element, `img`.

Listing 20-19: XHTML1-image.mod: the image module

```

<!-- ..... -->
<!-- XHTML 1.0 Images Module ..... -->
<!-- file: XHTML1-image.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved.
    Revision: @(#)XHTML1-image.mod 1.1599/04/01 SMI

    This DTD module is identified by the PUBLIC and SYSTEM
    identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Images//EN"
    SYSTEM "XHTML1-image.mod"

    Revisions:
# 1999-01-31 corrected transitional attributes (errata)
    ..... -->

<!-- d3.1. Images

    img

-->

<!-- To avoid problems with text-only UAs as well as
    to make image content understandable and navigable
    to users of non-visual UAs, you need to provide
    a description with ALT, and avoid server-side image maps
-->

<!ENTITY % Img.content "EMPTY" >
<!ELEMENT img %Img.content; >
<!ATTLIST img
    %Common.attrib;
    src          %URI;          #REQUIRED
    alt          %Text;         #REQUIRED
    longdesc    %URI;          #IMPLIED
    height      %Length;       #IMPLIED
    width       %Length;       #IMPLIED
    usemap      %URI;          #IMPLIED
    ismap       (ismap)        #IMPLIED

```

Continued

Listing 20-19 (continued)

```

>
<!-- USEMAP points to a MAP element which may be in this
      document or an external document, although the latter is
      not widely supported
-->

<![%XHTML.Transitional;[
<!-- additional Transitional attributes ->
<!ATTLIST img
      %IAAlign.attrib;
      border           %Pixels;           #IMPLIED
      hspace           %Pixels;           #IMPLIED
      vspace           %Pixels;           #IMPLIED
>
]]>

<!-- end of XHTML1-image.mod -->

```

Note that the `alt` attribute is required on `img`. Omitting it produces a validity error.

The Frames Module

Next, both the strict and transitional DTDs conditionally import the frames module, `XHTML1-frames.mod` shown in Listing 20-20. This module defines those elements and attributes used on Web pages with frames. Specifically, it defines the `frameset`, `frame`, `noframes`, and `iframe` elements and their associated attribute lists.

However, this import is wrapped in:

```

<![%XHTML1-frames.module;[
  <!-- frames declarations -->
]]>

```

Consequently, these imports only take place if `%XHTML1-frames.module;` parameter entity reference evaluates to `INCLUDE` which it does only if the `frameset` DTD is in use.

Listing 20-20: XHTML1-image.mod: the frames module

```

<!-- ..... ->
<!-- XHTML 1.0 Frames Module ..... ->
<!-- file: XHTML1-frames.mod

```

```

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved.
Revision: @(#)XHTML1-frames.mod 1.15 99/04/01 SMI

```

```

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

```

```

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Frames//EN"
SYSTEM "XHTML1-frames.mod"

```

```

Revisions:

```

```

#1999-01-14 transferred 'target' attribute on 'a' from linking
module

```

```

..... ->

```

```

<!-- a2. Frames

```

```

    frame, frameset, iframe, noframes

```

```

->

```

```

<!-- The content model for HTML documents depends on whether
the HEAD is followed by a FRAMESET or BODY element. The
widespread omission of the BODY start tag makes it
impractical to define the content model without the use of
a conditional section.

```

```

->

```

```

<!ENTITY % Frameset.content "(( frameset | frame )+, noframes?
)" >

```

```

<!ELEMENT frameset %Frameset.content; >

```

```

<!ATTLIST frameset

```

```

    %Core.attrib;

```

```

    rows %MultiLengths; #IMPLIED

```

```

    cols %MultiLengths; #IMPLIED

```

```

>

```

```

<!-- reserved frame names start with "_" otherwise starts with
letter ->

```

```

<!ENTITY % Frame.content "EMPTY" >

```

```

<!ELEMENT frame %Frame.content; >

```

```

<!ATTLIST frame

```

```

    %Core.attrib;

```

```

    longdesc %URI; #IMPLIED

```

```

    name CDATA #IMPLIED

```

```

    src %URI; #IMPLIED

```

```

    frameborder (1|0) '1'

```

```

    marginwidth %Pixels; #IMPLIED

```

```

    marginheight %Pixels; #IMPLIED

```

```

    noresize (noresize) #IMPLIED

```

```

    scrolling (yes|no|auto) 'auto'

```

Continued

Listing 20-20 (continued)

```

>
<!-- Inline Frames ..... ->
<!ENTITY % Iframe.content "( %Flow.mix; )*" >
<!ELEMENT iframe %Iframe.content; >
<!ATTLIST iframe
    %Core.attrib;
    longdesc      %URI;                #IMPLIED
    name          CDATA                #IMPLIED
    src           %URI;                #IMPLIED
    frameborder   (1|0)                '1'
    marginwidth   %Pixels;             #IMPLIED
    marginheight  %Pixels;             #IMPLIED
    scrolling      (yes|no|auto)        'auto'
    %IAAlign.attrib;
    height        %Length;             #IMPLIED
    width         %Length;             #IMPLIED
>

<!-- changes to other declarations ..... ->

<!-- redefine content model for html element,
      substituting frameset for body ->
<!ENTITY % Html.content "( head, frameset )" >

<!-- alternate content container for non frame-based rendering
->

<!ENTITY % Noframes.content "( body )">
<!-- in HTML 4.0 was "( body ) -( noframes )"
      exclusion on body ->
<!ELEMENT noframes %Noframes.content; >
<!ATTLIST noframes
    %Common.attrib;
>

<!-- add 'target' attribute to 'a' element ->
<!ATTLIST a
    target        %FrameTarget;        #IMPLIED
>

<!-- end of XHTML1-frames.mod ->

```

There's not a lot to say about these declarations. There are no particularly interesting tricks here you haven't seen before, and adding frames to the DTD doesn't require overriding any previous parameter entities, at least not here. The most unusual aspect of this particular module is that the name attribute of both `frame` and `iframe` appears as CDATA rather than as some parameter entity reference. The reason is that there aren't any significant restrictions on frame names other than that they be CDATA. An eventual schema language can't add anything to raw CDATA in this case.

The Linking Module

The next module imported by both strict and transitional DTDs, XHTML1-image.mod, shown in Listing 20-21, is another simple module that defines the linking elements `a`, `base`, and `link`.

Listing 20-21: XHTML1-image.mod: the linking module

```

<!-- ..... ->
<!-- XHTML 1.0 Linking Module ..... ->
<!-- file: XHTML1-linking.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved.
    Revision: @(#)XHTML1-linking.mod 1.13 99/04/01 SMI

    This DTD module is identified by the PUBLIC
    and SYSTEM identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Linking//EN"
    SYSTEM "XHTML1-linking.mod"

    Revisions:
    # 1998-10-27 exclusion on 'a' within 'a' removed for XML
    # 1998-11-15 moved shape and coords attributes on 'a' to
                  csismap module
    # 1999-01-14 moved onfocus and onblur attributes on 'a' to
                  events module
    ..... ->

<!-- d2. Linking

        a, base, link
->

<!-- ..... Anchor Element ..... ->
<!ENTITY % Shape "(rect|circle|poly|default)">

```

Continued

Listing 20-21 (continued)

```

<!ENTITY % Coords "CDATA" >

<!ENTITY % A.content "( #PCDATA | %Inline-noa.mix; )*" >
<!ELEMENT a %A.content; >
<!ATTLIST a
    %Common.attrib;
    name          CDATA                #IMPLIED
    href          %URI;                #IMPLIED
    %Alink.attrib;
    charset      %Charset;            #IMPLIED
    type         %ContentType;        #IMPLIED
    hreflang     %LanguageCode;       #IMPLIED
    rel          %LinkTypes;          #IMPLIED
    rev          %LinkTypes;          #IMPLIED
    accesskey    %Character;          #IMPLIED
    tabindex     %Number;             #IMPLIED
>

<!-- ..... Base Element ..... -->

<!ENTITY % Base.content "EMPTY" >
<!ELEMENT base %Base.content; >
<!ATTLIST base
    href          %URI;                #REQUIRED
>

<!-- ..... Link Element ..... -->

<!-- Relationship values can be used in principle:

    a) for document specific toolbars/menus when used
       with the LINK element in document head e.g.
       start, contents, previous, next, index, end, help
    b) to link to a separate style sheet (rel=stylesheet)
    c) to make a link to a script (rel=script)
    d) by stylesheets to control how collections of
       html nodes are rendered into printed documents
    e) to make a link to a printable version of this document
       e.g. a postscript or pdf version
       (rel=alternate media=print)
-->

<!ENTITY % Link.content "EMPTY" >
<!ELEMENT link %Link.content; >
<!ATTLIST link
    %Common.attrib;
    charset      %Charset;            #IMPLIED
    href          %URI;                #IMPLIED
    hreflang     %LanguageCode;       #IMPLIED
    type         %ContentType;        #IMPLIED
    rel          %LinkTypes;          #IMPLIED

```

```

        rev          %LinkTypes;          #IMPLIED
        media       %MediaDesc;          #IMPLIED
>
<!-- end of XHTML1-linking.mod -->

```

The Client-side Image Map Module

The next module imported by both strict and transitional DTDs, `XHTML1-csismap.mod`, shown in Listing 20-22, is another simple module that defines the client-side image map elements `map` and `area`. The `map` element provides a client-side image map and must contain one or more block-level elements, miscellaneous elements, or `area` elements. The `area` element has an unusual, non-standard set of attributes. This should not surprise you, though, because the `area` element is unlike most other HTML elements. It's the only HTML element that acts like a vector graphic.

Listing 20-22: XHTML1-csismap.mod: the client-side image map module

```

<!-- ..... -->
<!-- XHTML 1.0 Client-side Image Map Module
..... -->
<!-- file: XHTML1-csismap.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved.
    Revision: @(#)XHTML1-csismap.mod 1.15 99/04/01 SMI

    This DTD module is identified by the
    PUBLIC and SYSTEM identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Client-side Image Maps//EN"
    SYSTEM "XHTML1-csismap.mod"

    Revisions:
    # 1999-01-31 fixed map content model (errata)
    ..... -->

<!-- d3.2. Client-side Image Maps

        area, map
-->

```

Continued

Listing 20-22 (continued)

```

<!-- These can be placed in the same document or grouped in a
      separate document although this isn't widely supported -->

<!ENTITY % Map.content
      "(( %Heading.class; | %List.class; | %Block.mix; ) | area)+">
<!ELEMENT map %Map.content; >
<!ATTLIST map
      %Common.attrib;
      name          CDATA          #REQUIRED
>

<!ENTITY % Area.content "EMPTY" >
<!ELEMENT area %Area.content; >
<!ATTLIST area
      %Common.attrib;
      href          %URI;          #IMPLIED
      shape        %Shape;        'rect'
      coords       %Coords;       #IMPLIED
      nohref       (nohref)       #IMPLIED
      alt          %Text;         #REQUIRED
      tabindex     %Number;       #IMPLIED
      accesskey    %Character;    #IMPLIED
>

<!-- modify anchor (<a>) attribute definition list to
      allow for client-side image maps -->

<!ATTLIST a
      shape        %Shape;        'rect'
      coords       %Coords;       #IMPLIED
>

<!-- end of XHTML1-csismap.mod -->

```

The Object Element Module

The next module imported by both strict and transitional DTDs, XHTML1-object.mod, shown in Listing 20-23, is another simple module that defines the `object` and `param` elements used to embed non-HTML content such as Java applets, ActiveX controls, and so forth in Web pages.

Listing 20-23: XHTML1-object.mod: the object module

```

<!-- ..... -->
<!-- XHTML 1.0 External Inclusion Module
..... -->

```

```

<!-- file: XHTML1-object.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved.
    Revision: @(#)XHTML1-object.mod 1.16 99/04/01 SMI

    This DTD module is identified by the
    PUBLIC and SYSTEM identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Object Element//EN"
    SYSTEM "XHTML1-object.mod"

    Revisions:
# 1999-01-31  changed object's archive attr
               to allow for multiple URIs
# 1999-01-31  corrected transitional attributes (errata)
               ..... ->

<!-- d3.3. Objects

        object, param

        object is used to embed objects as part of HTML pages;
        param elements should precede other content.
->

<!ENTITY % Object.content "( %Flow.mix; | param )*" >
<!ELEMENT object %Object.content; >
<!ATTLIST object
    %Common.attrib;
    declare          (declare)          #IMPLIED
    classid          %URI;              #IMPLIED
    codebase         %URI;              #IMPLIED
    data             %URI;              #IMPLIED
    type             %ContentType;      #IMPLIED
    codetype         %ContentType;      #IMPLIED
    archive          %URIs;             #IMPLIED
    standby          %Text;             #IMPLIED
    height           %Length;           #IMPLIED
    width           %Length;           #IMPLIED
    usemap          %URI;              #IMPLIED
    name            CDATA               #IMPLIED
    tabindex        %Number;           #IMPLIED
>

<![%XHTML.Transitional;[
<!-- additional Transitional attributes ->
<!ATTLIST object
    %IAlign.attrib;
    border           %Pixels;          #IMPLIED
    hspace          %Pixels;          #IMPLIED

```

Continued

Listing 20-23 (continued)

```

        vspace          %Pixels;                #IMPLIED
    >
]]>

<!ENTITY % Param.content "EMPTY" >
<!ELEMENT param %Param.content; >
<!ATTLIST param
    id          ID                #IMPLIED
    name        CDATA             #REQUIRED
    value       CDATA             #IMPLIED
    valuetype   (data|ref|object) 'data'
    type        %ContentType;     #IMPLIED
>

<!-- end of XHTML1-object.mod -->

```

Only two elements are declared; `object` and `param`. The content model for `object` is spelled out using the `Flow.mix` and `param` entities. Also, note that the mixed-content model of the `object` element requires a stricter declaration than is actually provided. That's the purpose of the comment "param elements should precede other content". However, a DTD can't specify that `param` elements should precede other content since mixed content requires that `#PCDATA` come first, and that a choice be used instead of a sequence.

The Java Applet Element Module

The `applet` element was originally invented by Sun to embed Java applets in Web pages. The next module imported only by the transitional DTD—`XHTML1-applet.mod`, shown in Listing 20-24—is another simple module that defines the `applet` element. However, HTML 4.0 deprecates the `applet` element in favor of the more generic `object` element which can embed not only applets, but also ActiveX controls, images, Shockwave animations, QuickTime movies, and other forms of active and multimedia content. Consequently, only the transitional XHTML DTD uses the `applet` module.

Listing 20-24: XHTML1-applet.mod: the applet module

```

<!-- ..... ->
<!-- XHTML 1.0 Draft Document Java Applet Module
..... ->
<!-- file: XHTML1-applet.mod

```

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights

```

Reserved.
Revision: @(#)XHTML1-applet.mod 1.14 99/04/01 SMI

This DTD module is identified by the
PUBLIC and SYSTEM identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML V1.0 Java Applets//EN"
SYSTEM "XHTML1-applet.mod"

Revisions:
(none)
..... ->

<!-- d4. Scripting

      applet
->

<!-- One of code or object attributes must be present.
      Place param elements before other content.
->

<!ENTITY % Applet.content "( param | %Flow.mix; )*">
<!ELEMENT applet %Applet.content; >
<!ATTLIST applet
      %Core.attrib;
      codebase      %URI;                #IMPLIED
      archive       CDATA                 #IMPLIED
      code          CDATA                 #IMPLIED
      object        CDATA                 #IMPLIED
      alt           %Text;                #IMPLIED
      name          CDATA                 #IMPLIED
      width         %Length;              #REQUIRED
      height        %Length;              #REQUIRED
      %IAalign.attrib;
      hspace        %Pixels;              #IMPLIED
      vspace        %Pixels;              #IMPLIED
>

<!-- If the Object module that supplies the param element
      declarations is not used, redeclare %Param.local.module;
      as 'INCLUDE': ->
<!ENTITY % Param.local.module "IGNORE" >
<![%Param.local.module;[
<!ENTITY % Param.content "EMPTY">
<!ELEMENT param %Param.content; >
<!ATTLIST param
      id            ID                    #IMPLIED
      name          CDATA                 #REQUIRED
      value         CDATA                 #IMPLIED
      valuetype     (data|ref|object)     'data'
      type          %ContentType;         #IMPLIED

```

Continued

Listing 20-24 (continued)

```
>
]]>

<!-- end of XHTML1-applet.mod -->
```

The content model and attribute list for `applet` essentially resembles `object`. The `param` element that's used to pass parameters to applets is declared in Listing 22-3, `XHTML1-object.mod`. However, if for some reason that's not imported as well, then the `Param.local.module` entity can be redefined to `INCLUDE` instead of `IGNORE`, and this DTD will declare `param`.

The Lists Module

The `XHTML1-list.mod` module, shown in Listing 20-25, operates in both DTDs and defines the elements used in ordered, unordered, and definition lists.

Listing 20-25: XHTML1-list.mod: the Voyager module for lists

```
<!-- ..... ->
<!-- XHTML 1.0 Lists Module
..... ->
<!-- file: XHTML1-list.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved.
Revision: @(#)XHTML1-list.mod 1.13 99/04/01 SMI

This DTD module is identified by the PUBLIC and SYSTEM
identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Lists//EN"
SYSTEM "XHTML1-list.mod"

Revisions:
(none)
..... ->

<!-- a3. Lists

    dl, dt, dd, ol, ul, li

A conditional section includes additional declarations for
the Transitional DTD
```

```

    dir, menu
->
<!-- definition lists - DT for term, DD for its definition ->
<!ENTITY % D1.content "( dt | dd )+" >
<!ELEMENT dl %D1.content; >
<!ATTLIST dl
    %Common.attrib;
>

<!ENTITY % Dt.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT dt %Dt.content; >
<!ATTLIST dt
    %Common.attrib;
>

<!ENTITY % Dd.content "( #PCDATA | %Flow.mix; )*" >
<!ELEMENT dd %Dd.content; >
<!ATTLIST dd
    %Common.attrib;
>

<!-- Ordered Lists (ol) numbered styles ->
<!ENTITY % O1.content "( li )+" >
<!ELEMENT ol %O1.content; >
<!ATTLIST ol
    %Common.attrib;
>

<!-- Unordered Lists (ul) bullet styles ->
<!ENTITY % U1.content "( li )+" >
<!ELEMENT ul %U1.content; >
<!ATTLIST ul
    %Common.attrib;
>

<!ENTITY % Li.content "( #PCDATA | %Flow.mix; )*" >
<!ELEMENT li %Li.content; >
<!ATTLIST li
    %Common.attrib;
>

<![%XHTML.Transitional;[
<!-- Ordered lists (ol) Numbering style

    1   arabic numbers      1, 2, 3, ...
    a   lower alpha        a, b, c, ...
    A   upper alpha        A, B, C, ...
    i   lower roman        i, ii, iii, ...

```

Continued

Listing 20-25 (continued)

```

    I    upper roman          I, II, III, ...

    The style is applied to the sequence number which by
    default is reset to 1 for the first list item in
    an ordered list.
->

<!ENTITY % O1Style "CDATA" >

<!ATTLIST ol
    type          %O1Style;          #IMPLIED
    compact      (compact)          #IMPLIED
    start        %Number;           #IMPLIED
>

<!-- Unordered Lists (ul) bullet styles -->
<!ENTITY % U1Style "(disc|square|circle)" >

<!ATTLIST ul
    type          %U1Style;          #IMPLIED
    compact      (compact)          #IMPLIED
>

<!ENTITY % Dir.content "( li )+" >
<!ELEMENT dir %Dir.content; >
<!ATTLIST dir
    %Common.attrib;
    compact      (compact)          #IMPLIED
>

<!ENTITY % Menu.content "( li )+" >
<!ELEMENT menu %Menu.content; >
<!ATTLIST menu
    %Common.attrib;
    compact      (compact)          #IMPLIED
>
]]>

<!-- end of XHTML1-list.mod -->

```

You can define ordered and unordered lists much the same way. Each contains one list element (`ol` or `ul`) which may contain one or more list items (`li`). Both `ol` and `ul` elements may have the standard `%Common.attrib`; attributes of any HTML element. The definition list resembles this except that `dl` `dt` pairs are used instead of `li` list items.

The Forms Module

The XHTML1-form.mod module — shown in Listing 20-26 and used in both DTDs — covers the standard HTML form elements form, label, input, select, optgroup, option, textarea, fieldset, legend, and button. This is a relatively complicated module, reflecting the complexity of HTML forms.

Listing 20-26: XHTML1-form.mod: the XHTML forms module

```

<!-- ..... ->
<!-- XHTML 1.0 Forms Module
..... ->
<!-- file: XHTML1-form.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved.
    Revision: @(#)XHTML1-form.mod 1.18 99/04/01 SMI

    This DTD module is identified by the PUBLIC and SYSTEM
    identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Forms//EN"
    SYSTEM "XHTML1-form.mod"

    Revisions:
    # 1998-10-27 exclusion on form within form removed for XML
    # 1998-11-10 changed button content model to mirror exclusions
    # 1999-01-31 added 'accept' attribute on form (errata)
    ..... ->

<!-- d7. Forms

    form, label, input, select, optgroup, option, textarea,
    fieldset, legend, button
->

<![%XHTML.Transitional;[
<!ENTITY % Form.content
    "( %Heading.class;
    | %List.class;
    | %Inline.class;
    | %Block-noform.mix;
    | fieldset )*"
>
]]>
<!ENTITY % Form.content
    "( %Heading.class;
    | %List.class;

```

Continued

Listing 20-26 (continued)

```

        | %Block-noform.mix;
        | fieldset )+"
>

<!ELEMENT form %Form.content; >
<!ATTLIST form
%Common.attrib;
action      %URI;          #REQUIRED
method      (get|post)     'get'
enctype     %ContentType;  'application/x-www-form-urlencoded'
accept-charset %Charsets;  #IMPLIED
accept      %ContentTypes; #IMPLIED
>

<!-- Each label must not contain more than ONE field -->

<!ENTITY % Label.content
        "( #PCDATA
        | %Inlstruct.class;
        | %Inlpres.class;
        | %Inlphras.class;
        | %Inlspecial.class;
        | input | select | textarea | button
        | %Misc.class; )*"
>
<!ELEMENT label %Label.content; >
<!ATTLIST label
%Common.attrib;
for          IDREF          #IMPLIED
accesskey   %Character;    #IMPLIED
>

<!ENTITY % InputType.class
        "( text | password | checkbox | radio | submit
        | reset | file | hidden | image | button )"
>

<!-- attribute name required for all but submit & reset -->

<!ENTITY % Input.content "EMPTY" >
<!ELEMENT input %Input.content; >
<!ATTLIST input
%Common.attrib;
type        %InputType.class;  'text'
name        CDATA              #IMPLIED
value       CDATA              #IMPLIED
checked     (checked)          #IMPLIED
disabled    (disabled)         #IMPLIED
readonly    (readonly)        #IMPLIED
size        CDATA              #IMPLIED
maxlength   %Number;          #IMPLIED

```

```

        src          %URI;                #IMPLIED
        alt          CDATA                #IMPLIED
        usemap       %URI;                #IMPLIED
        tabindex     %Number;             #IMPLIED
        accesskey    %Character;          #IMPLIED
        accept       %ContentTypes;       #IMPLIED
    >

<!ENTITY % Select.content "( optgroup | option )+" >
<!ELEMENT select %Select.content; >
<!ATTLIST select
    %Common.attrib;
    name          CDATA                #IMPLIED
    size          %Number;             #IMPLIED
    multiple      (multiple)           #IMPLIED
    disabled      (disabled)           #IMPLIED
    tabindex     %Number;             #IMPLIED
>

<!ENTITY % Optgroup.content "( option )+" >
<!ELEMENT optgroup %Optgroup.content; >
<!ATTLIST optgroup
    %Common.attrib;
    disabled      (disabled)           #IMPLIED
    label         %Text;                #REQUIRED
>

<!ENTITY % Option.content "( #PCDATA )" >
<!ELEMENT option %Option.content; >
<!ATTLIST option
    %Common.attrib;
    selected      (selected)           #IMPLIED
    disabled      (disabled)           #IMPLIED
    label         %Text;                #IMPLIED
    value         CDATA                 #IMPLIED
>

<!ENTITY % Textarea.content "( #PCDATA )" >
<!ELEMENT textarea %Textarea.content; >
<!ATTLIST textarea
    %Common.attrib;
    name          CDATA                #IMPLIED
    rows          %Number;             #REQUIRED
    cols          %Number;             #REQUIRED
    disabled      (disabled)           #IMPLIED
    readonly      (readonly)           #IMPLIED
    tabindex     %Number;             #IMPLIED
    accesskey     %Character;          #IMPLIED
>

<!-- #PCDATA is to solve the mixed content problem, per
specification only whitespace is allowed there!

```

Continued

Listing 20-26 (continued)

```

->

<!ENTITY % Fieldset.content
    "( #PCDATA | legend | %Flow.mix; )*" >
<!ELEMENT fieldset %Fieldset.content; >
<!ATTLIST fieldset
    %Common.attrib;
>

<![%XHTML.Transitional;[
<!ENTITY % LegendAlign.attrib
    "align          (top|bottom|left|right) #IMPLIED" >
]]>
<!ENTITY % LegendAlign.attrib "" >

<!ENTITY % Legend.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT legend %Legend.content; >
<!ATTLIST legend
    %Common.attrib;
    accesskey %Character;          #IMPLIED
    %LegendAlign.attrib;
>

<!ENTITY % Button.content
    "( #PCDATA
    | %Heading.class;
    | %List.class;
    | %Inlpres.class;
    | %Inlphras.class;
    | %Block-noform.mix;
    | img | object | map )" >
>
<!ELEMENT button %Button.content; >
<!ATTLIST button
    %Common.attrib;
    name          CDATA          #IMPLIED
    value         CDATA          #IMPLIED
    type          (button|submit|reset) 'submit'
    disabled      (disabled)     #IMPLIED
    tabindex      %Number;       #IMPLIED
    accesskey     %Character;     #IMPLIED
>

<!-- end of forms.mod -->

```

This module is starting to come close to the limits of DTDs. Several times you see comments specifying restrictions that are difficult to impossible to include in the declarations. For example, the comment that “attribute name required for all but submit & reset” for `input` elements. You can specify that all `input` elements must have a `name` attribute, or you can specify that all `input` elements may or may not have a `name` attribute, but you cannot specify that some must have it while others do not have to have it.

You might argue that this points more toward a deficiency in HTML forms than a deficiency in DTDs, and perhaps you’d be right. After all, submit and reset buttons certainly don’t have to be `input` elements. Still, you can witness several other places in this module where the DTD begins to creak under its own weight. Perhaps what’s really being demonstrated here is that XML and DTDs were designed for display of static documents, not for heavy interactive use.

The Table Module

The `XHTML1-table.mod` module, shown in Listing 20-15 and used by both DTDs, defines the elements used to lay out tables in HTML; specifically `caption`, `col`, `colgroup`, `table`, `tbody`, `td`, `tfoot`, `th`, `thead`, and `tr`. Like form elements, most of these elements should only appear inside a `table` element and consequently this module runs somewhat longer since it can’t rely on elements defined previously, and since many elements defined here don’t appear anywhere else.

Listing 20-27: XHTML1-table.mod: the XHTML tables module

```
<!-- ..... ->
<!-- XHTML 1.0 Table Module
..... ->
<!-- file: XHTML1-table.mod

This is XHTML 1.0, an XML reformulation of HTML 4.0.
Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
Reserved.
Revision: @(#)XHTML1-table.mod 1.15 99/04/01 SMI

This DTD module is identified by the
PUBLIC and SYSTEM identifiers:

PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Tables//EN"
SYSTEM "XHTML1-table.mod"

Revisions:
(none)
..... ->

<!-- d6. HTML 4.0 Tables
```

Continued

Listing 20-27 (continued)

```
caption, col, colgroup, table, tbody,
td, tfoot, th, thead, tr
```

A conditional section includes additional declarations for the Transitional DTD

→

```
<!-- IETF HTML table standard, see [RFC1942] -->
```

```
<!-- The border attribute sets the thickness of the frame
around the table. The default units are screen pixels.
```

The frame attribute specifies which parts of the frame around the table should be rendered. The values are not the same as CALS to avoid a name clash with the valign attribute.

The value "border" is included for backwards compatibility with <table border> which yields frame=border and border=implied For <table border="1"> you get border="1" and frame="implied". In this case, it is appropriate to treat this as frame=border for backwards compatibility with deployed browsers.

→

```
<!ENTITY % TFrame
"(void|above|below|hsides|lhs|rhs|vsides|box|border)">
```

```
<!-- The rules attribute defines which rules to draw between
cells:
```

If rules is absent then assume:

"none" if border is absent or border="0" otherwise "all"

→

```
<!ENTITY % TRules "(none | groups | rows | cols | all)">
```

```
<!-- horizontal placement of table relative to document -->
```

```
<!ENTITY % TAlign "(left|center|right)">
```

```
<!-- horizontal alignment attributes for cell contents -->
```

```
<!ENTITY % CellHAlign.attrib
"align      (left|center|right|justify|char) #IMPLIED
char        %Character;                    #IMPLIED
charoff     %Length;                       #IMPLIED"
>
```

```
<!-- vertical alignment attributes for cell contents -->
```

```
<!ENTITY % CellVAlign.attrib
```

```

    "valign      (top|middle|bottom|baseline) #IMPLIED"
>
<!ENTITY % CaptionAlign "(top|bottom|left|right)">
<!-- Scope is simpler than axes attribute for common tables -->
<!ENTITY % Scope "(row|col|rowgroup|colgroup)" >
<!ENTITY % Table.content
    "( caption?, ( col* | colgroup* ),
     (( thead?, tfoot?, tbody+ ) | ( tr+ )))"
>
<!ELEMENT table %Table.content; >
<!ATTLIST table
    %Common.attrib;
    summary      %Text;                #IMPLIED
    width        %Length;              #IMPLIED
    border       %Pixels;              #IMPLIED
    frame        %TFrame;              #IMPLIED
    rules        %TRules;              #IMPLIED
    cellspacing  %Length;              #IMPLIED
    cellpadding %Length;              #IMPLIED
    datapagesize CDATA                  #IMPLIED
>

<!ENTITY % Caption.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT caption %Caption.content; >
<!ATTLIST caption
    %Common.attrib;
>

<!ENTITY % Thead.content "( tr )+" >
<!ELEMENT thead %Thead.content; >
<!-- Use thead to duplicate headers when breaking table
      across page boundaries, or for static headers when
      TBODY sections are rendered in scrolling panel.

      Use tfoot to duplicate footers when breaking table
      across page boundaries, or for static footers when
      TBODY sections are rendered in scrolling panel.

      Use multiple tbody sections when rules are needed
      between groups of table rows.
-->
<!ATTLIST thead
    %Common.attrib;
    %CellHAlign.attrib;
    %CellVAlign.attrib;
>

<!ENTITY % Tfoot.content "( tr )+" >

```

Continued

Listing 20-27 (continued)

```

<!ELEMENT tfoot %Tfoot.content; >
<!ATTLIST tfoot
    %Common.attrib;
    %CellHAlign.attrib;
    %CellVAlign.attrib;
>

<!ENTITY % Tbody.content "( tr )+" >
<!ELEMENT tbody %Tbody.content; >
<!ATTLIST tbody
    %Common.attrib;
    %CellHAlign.attrib;
    %CellVAlign.attrib;
>

<!ENTITY % Colgroup.content "( col )*" >
<!ELEMENT colgroup %Colgroup.content; >
<!-- colgroup groups a set of col elements. It allows you to
    group several semantically related columns together.
-->
<!ATTLIST colgroup
    %Common.attrib;
    span %Number; '1'
    width %MultiLength; #IMPLIED
    %CellHAlign.attrib;
    %CellVAlign.attrib;
>

<!ENTITY % Col.content "EMPTY" >
<!ELEMENT col %Col.content; >
<!-- col elements define the alignment properties for cells in
    one or more columns.

    The width attribute specifies the width of the columns, e.g.

        width="64"           width in screen pixels
        width="0.5*"        relative width of 0.5

    The span attribute causes the attributes of one
    col element to apply to more than one column.
-->
<!ATTLIST col
    %Common.attrib;
    span %Number; '1'
    width %MultiLength; #IMPLIED
    %CellHAlign.attrib;
    %CellVAlign.attrib;
>

<!ENTITY % Tr.content "( th | td )+" >
<!ELEMENT tr %Tr.content; >

```

```

<!ATTLIST tr
    %Common.attrib;
    %CellHAlign.attrib;
    %CellVAlign.attrib;
>

<!-- th is for headers, td for data, but for cells
    acting as both use td -->

<!ENTITY % Th.content "( #PCDATA | %Flow.mix; )*" >
<!ELEMENT th %Th.content; >
<!ATTLIST th
    %Common.attrib;
    abbr %Text; #IMPLIED
    axis CDATA #IMPLIED
    headers IDREFS #IMPLIED
    scope %Scope; #IMPLIED
    rowspan %Number; '1'
    colspan %Number; '1'
    %CellHAlign.attrib;
    %CellVAlign.attrib;
>

<!ENTITY % Td.content "( #PCDATA | %Flow.mix; )*" >
<!ELEMENT td %Td.content; >
<!ATTLIST td
    %Common.attrib;
    abbr %Text; #IMPLIED
    axis CDATA #IMPLIED
    headers IDREFS #IMPLIED
    scope %Scope; #IMPLIED
    rowspan %Number; '1'
    colspan %Number; '1'
    %CellHAlign.attrib;
    %CellVAlign.attrib;
>

<![%XHTML.Transitional;[
<!-- additional Transitional attributes for XHTML tables:
    (in XML, multiple ATTLIST declarations are merged)
-->

<!ATTLIST table
    align %TAlign; #IMPLIED
    bgcolor %Color; #IMPLIED
>

<!ATTLIST caption
    align %CaptionAlign; #IMPLIED
>

<!ATTLIST tr

```

Continued

Listing 20-27 (continued)

```

    bgcolor      %Color;                #IMPLIED
  >

  <!ATTLIST th
    nowrap      (nowrap)                #IMPLIED
    bgcolor     %Color;                  #IMPLIED
    width       %Pixels;                 #IMPLIED
    height      %Pixels;                 #IMPLIED
  >

  <!ATTLIST td
    nowrap      (nowrap)                #IMPLIED
    bgcolor     %Color;                  #IMPLIED
    width       %Pixels;                 #IMPLIED
    height      %Pixels;                 #IMPLIED
  >
]]>

<!-- end of XHTML1-table.mod -->

```

The Meta Module

The next module is imported by both strict and transitional DTDs. **XHTML1-meta.mod**, shown in Listing 20-28, gets its name by defining the `meta` element placed in HTML `head` elements to provide keyword, authorship, abstract, and other indexing information that's mostly useful to Web robots. This module also defines the `title` element. Although the `title` is meta-information in some sense, I suspect **XHTML1-head.mod** might be a better name here, except that the `head` element isn't defined here.

Listing 20-28: XHTML1-meta.mod: the XHTML meta module

```

<!-- ..... ->
<!-- XHTML 1.0 Document Metainformation Module
..... ->
<!-- file: XHTML1-meta.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio), All Rights
    Reserved.
    Revision: @(#)XHTML1-meta.mod 1.14 99/04/01 SMI

```

```

    This DTD module is identified by the PUBLIC and SYSTEM
    identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Metainformation//EN"
    SYSTEM "XHTML1-meta.mod"

    Revisions:
    # 1998-11-11 title content model changed
                - exclusions no longer necessary
    # 1999-02-01 removed isindex
                ..... ->

<!-- d1. Meta Information

    meta, title
->

<!-- The title element is not considered part of the flow of
    text. It should be displayed, for example as the page
    header or window title. Exactly one title is required per
    document.
->

<!ENTITY % Title.content "( #PCDATA )" >
<!ELEMENT title %Title.content; >
<!ATTLIST title
    %I18n.attrib;
>

<!ENTITY % Meta.content "EMPTY" >
<!ELEMENT meta %Meta.content; >
<!ATTLIST meta
    %I18n.attrib;
    http-equiv    NMTOKEN          #IMPLIED
    name          NMTOKEN          #IMPLIED
    content       CDATA            #REQUIRED
    scheme        CDATA            #IMPLIED
>

<!-- end of XHTML1-meta.mod ->

```

The Structure Module

The final standard module takes all the previously defined elements, attributes, and entities and puts them together in an HTML document. This is XHTML1-struct.mod, shown in Listing 20-29. Specifically, it defines the `html`, `head`, and `body` elements.

Listing 20-29: XHTML1-struct.mod: the XHTML structure module

```

<!-- ..... ->
<!-- XHTML 1.0 Structure Module
..... ->
<!-- file: XHTML1-struct.mod

    This is XHTML 1.0, an XML reformulation of HTML 4.0.
    Copyright 1998-1999 W3C (MIT, INRIA, Keio),
    All Rights Reserved.
    Revision: @(#)XHTML1-struct.mod 1.15 99/04/01 SMI

    This DTD module is identified by the PUBLIC and SYSTEM
    identifiers:

    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Document Structure//EN"
    SYSTEM "XHTML1-struct.mod"

    Revisions:
# 1998-10-27 content model on head changed to
              exclude multiple title or base
# 1998-11-11 ins and del inclusions on body removed,
              added to indiv. elements
# 1998-11-15 added head element version attribute
              (restoring from HTML 3.2)
# 1999-03-24 %Profile.attrib; unused,
              but reserved for future use
              ..... ->

<!-- a1. Document Structure

        body, head, html
->

<!ENTITY % Head-opts.mix "( script | style | meta | link |
object )" * >

<!ENTITY % Head.content "( title, base?, %Head-opts.mix; )" >

<!-- reserved for future use with document profiles ->
<!ENTITY % Profile.attrib
        "profile %URI; #FIXED
'%XHTML.profile;'" >

<!ELEMENT head %Head.content; >
<!ATTLIST head
        %I18n.attrib;
        profile %URI; #IMPLIED
>

<![%XHTML.Transitional;[

```

```

<!-- in Transitional, allow #PCDATA and inlines directly within
body ->

<!ENTITY % Body.content "( #PCDATA | %Flow.mix; )*" >
]]>
<!ENTITY % Body.content
    "( %Heading.class;
      | %List.class;
      | %Block.class;
      | %Misc.class; )+"
>

<!ELEMENT body %Body.content; >
<!ATTLIST body
    %Common.attrib;
>

<![%XHTML.Transitional;[
<!-- .... additional Transitional attributes on body .... ->

<!-- There are also 16 widely known color names with their sRGB
values:

Black =#000000 Maroon =#800000 Green = #008000 Navy = #000080
Silver=#C0C0C0 Red =#FF0000 Lime = #00FF00 Blue = #0000FF
Gray =#808080 Purple =#800080 Olive = #808000 Teal = #008080
White =#FFFFFF Fuchsia=#FF00FF Yellow = #FFFF00 Aqua = #00FFFF
->

<!ATTLIST body
    bgcolor %Color; #IMPLIED
    text %Color; #IMPLIED
    link %Color; #IMPLIED
    vlink %Color; #IMPLIED
    alink %Color; #IMPLIED
    background %URI; #IMPLIED
>
]]>

<!ENTITY % Html.content "( head, body )" >

<!--version and namespace attribute values defined in driver->
<!ENTITY % Version.attrib
    "version CDATA #FIXED '%HTML.version;' " >
<!ENTITY % Ns.attrib
    "xmlns %URI; #FIXED '%XHTML.ns;' " >

<!ELEMENT html %Html.content; >
<!ATTLIST html
    %I18n.attrib;
    %Version.attrib;
    %Ns.attrib;

```

Continued

Listing 20-29 (continued)

```
>  
  
<!-- end of XHTML1-struct.mod -->
```

Non-Standard modules

There are a number of non-standard modules included in the XHTML distribution that aren't used as part of the main XHTML application and won't be discussed here, but may be useful as parts of your custom program. These include:

- ♦ XHTML1-form32.mod: HTML 3.2 forms (as opposed to the HTML 4.0 forms used by XHTML)
- ♦ XHTML1-table32.mod: HTML 3.2 tables (as opposed to the HTML 4.0 tables used by XHTML)
- ♦ XHTML1-math.mod: MathML with slight revisions to make it fully compatible with XHTML

The XHTML Entity Sets

XML requires all entities to be declared (with the possible exception of the five standard entity references `<`, `>`, `'`, `"`, and `&`). The XHTML DTD defines three entity sets declaring all entities commonly used in HTML:

1. XHTML1-lat1.ent, characters 160 through 255 of Latin-1, Listing 20-30.
2. XHTML1-symbol.ent, assorted useful characters and punctuation marks from outside the Latin-1 set such as the Euro sign and the em dash, Listing 20-31.
3. XHTML1-special.ent, the Greek alphabet and assorted symbols commonly used for math like ∞ and \int , Listing 20-32.

Each of these entity sets is included in all versions of the XHTML DTD through the XHTML1-chars.mod module. Each of these entity sets has the same basic format:

1. A comment containing basic title, usage, and copyright information.
2. Lots of general internal entity declarations. The value of each general entity is given as a character reference to a Unicode character. Since no one can be expected to remember the all 40,000 Unicode characters by number, a brief textual description of the referenced character is given in a comment following each entity declaration.

The XHTML Latin-1 Entities

The XHTML1-lat1.ent file shown in Listing 20-30 declares entity references for the upper half of the ISO 8859-1, Latin-1 character set.

Listing 20-30: XHTML1-lat1.ent: the XHTML entity set for the upper half of ISO 8859-1, Latin-1

```
<!-- XML-compatible ISO Latin 1 Character Entity Set for XHTML
1.0
```

Typical invocation:

```
<!ENTITY % XHTML1-lat1
PUBLIC "-//W3C//ENTITIES Latin 1//EN//XML"
"XHTML1-lat1.ent">
%XHTML1-lat1;
```

Revision: @(#)XHTML1-lat1.ent 1.13 99/04/01 SMI

Portions (C) International Organization for Standardization 1986 Permission to copy in any form is granted for use with conforming SGML systems and applications as defined in ISO 8879, provided this notice is included in all copies.

```
->
<!ENTITY nbsp "&#160;" ><!-- no-break space=non-breaking space,
U+00A0 ISOnum ->
<!ENTITY iexcl "&#161;" ><!-- inverted exclamation mark,
U+00A1 ISOnum ->
<!ENTITY cent "&#162;" ><!-- cent sign,
U+00A2 ISOnum ->
<!ENTITY pound "&#163;" ><!-- pound sign,
U+00A3 ISOnum ->
<!ENTITY curren "&#164;" ><!-- currency sign,
U+00A4 ISOnum ->
<!ENTITY yen "&#165;" ><!-- yen sign = yuan sign,
U+00A5 ISOnum ->
<!ENTITY brvbar "&#166;" ><!-- broken bar =broken vertical bar,
U+00A6 ISOnum ->
<!ENTITY sect "&#167;" ><!-- section sign,
U+00A7 ISOnum ->
<!ENTITY uml "&#168;" ><!-- diaeresis = spacing diaeresis,
U+00A8 ISODia ->
<!ENTITY copy "&#169;" ><!-- copyright sign,
U+00A9 ISOnum ->
<!ENTITY ordf "&#170;" ><!-- feminine ordinal indicator,
U+00AA ISOnum ->
<!ENTITY laquo "&#171;" ><!-- left-pointing double angle
```

Continued

Listing 20-30 (continued)

```

        quotation mark = left pointing guillemet,
                        U+00AB ISOnum ->
<!ENTITY not    "&#172;" ><!-- not sign,
                        U+00AC ISOnum ->
<!ENTITY shy   "&#173;" ><!-- soft hyphen = discretionary hyphen,
                        U+00AD ISOnum ->
<!ENTITY reg    "&#174;" ><!-- registered sign
                        = registered trade mark sign,
                        U+00AE ISOnum ->
<!ENTITY macr   "&#175;" ><!-- macron = spacing macron
                        = overline = APL overbar,
                        U+00AF ISODia ->
<!ENTITY deg    "&#176;" ><!-- degree sign,
                        U+00B0 ISOnum ->
<!ENTITY plusmn "&#177;" ><!-- plus-minus sign
                        = plus-or-minus sign,
                        U+00B1 ISOnum ->
<!ENTITY sup2   "&#178;" ><!-- superscript two
                        = superscript digit two = squared,
                        U+00B2 ISOnum ->
<!ENTITY sup3   "&#179;" ><!-- superscript three
                        = superscript digit three = cubed,
                        U+00B3 ISOnum ->
<!ENTITY acute  "&#180;" ><!-- acute accent = spacing acute,
                        U+00B4 ISODia ->
<!ENTITY micro  "&#181;" ><!-- micro sign,
                        U+00B5 ISOnum ->
<!ENTITY para   "&#182;" ><!-- pilcrow sign = paragraph sign,
                        U+00B6 ISOnum ->
<!ENTITY middot "&#183;" ><!-- middle dot = Georgian comma
                        = Greek middle dot,
                        U+00B7 ISOnum ->
<!ENTITY cedil  "&#184;" ><!-- cedilla = spacing cedilla,
                        U+00B8 ISODia ->
<!ENTITY sup1   "&#185;" ><!-- superscript one
                        = superscript digit one,
                        U+00B9 ISOnum ->
<!ENTITY ordm   "&#186;" ><!-- masculine ordinal indicator,
                        U+00BA ISOnum ->
<!ENTITY raquo  "&#187;" ><!-- right-pointing
        double angle quotation mark = right pointing guillemet,
                        U+00BB ISOnum ->
<!ENTITY frac14 "&#188;" ><!-- vulgar fraction one quarter
                        = fraction one quarter,
                        U+00BC ISOnum ->
<!ENTITY frac12 "&#189;" ><!-- vulgar fraction one half
                        = fraction one half,
                        U+00BD ISOnum ->
<!ENTITY frac34 "&#190;" ><!-- vulgar fraction three quarters
                        = fraction three quarters,
                        U+00BE ISOnum ->

```

```

<!ENTITY iquest "¿" ><!-- inverted question mark
                        = turned question mark,
                        U+00BF ISOnum ->
<!ENTITY Agrave "À" ><!--latin capital letter A with grave
                        = latin capital letter A grave,
                        U+00C0 ISOlat1 ->
<!ENTITY Aacute "Á"><!--latin capital letter A with acute,
                        U+00C1 ISOlat1 ->
<!ENTITY Acirc  "Â" ><!-- latin capital letter A
                        with circumflex,
                        U+00C2 ISOlat1 ->
<!ENTITY Atilde "Ã"><!--latin capital letter A with tilde,
                        U+00C3 ISOlat1 ->
<!ENTITY Auml  "Ä" ><!-- latin capital letter A
                        with diaeresis,
                        U+00C4 ISOlat1 ->
<!ENTITY Aring  "Å" ><!-- latin capital letter A
                        with ring above
                        = latin capital letter A ring,
                        U+00C5 ISOlat1 ->
<!ENTITY AElig  "Æ" ><!-- latin capital letter AE
                        = latin capital ligature AE,
                        U+00C6 ISOlat1 ->
<!ENTITY Ccedil "Ç" ><!-- latin capital letter C
                        with cedilla,
                        U+00C7 ISOlat1 ->
<!ENTITY Egrave "È"><!--latin capital letter E with grave,
                        U+00C8 ISOlat1 ->
<!ENTITY Eacute "É"><!--latin capital letter E with acute,
                        U+00C9 ISOlat1 ->
<!ENTITY Ecirc  "Ê" ><!-- latin capital letter E
                        with circumflex,
                        U+00CA ISOlat1 ->
<!ENTITY Euml  "Ë" ><!-- latin capital letter E with
diaeresis,
                        U+00CB ISOlat1 ->
<!ENTITY Igrave "Ì"><!--latin capital letter I with grave,
                        U+00CC ISOlat1 ->
<!ENTITY Iacute "Í"><!--latin capital letter I with acute,
                        U+00CD ISOlat1 ->
<!ENTITY Icirc  "Î" ><!-- latin capital letter I
                        with circumflex,
                        U+00CE ISOlat1 ->
<!ENTITY Iuml  "Ï" ><!-- latin capital letter I
                        with diaeresis,
                        U+00CF ISOlat1 ->
<!ENTITY ETH   "Ð" ><!-- latin capital letter ETH,
                        U+00D0 ISOlat1 ->
<!ENTITY Ntilde "Ñ"><!--latin capital letter N with tilde,
                        U+00D1 ISOlat1 ->
<!ENTITY Ograve "Ò"><!--latin capital letter O with grave,
                        U+00D2 ISOlat1 ->

```

Continued


```

                                U+00E6 ISOlat1 ->
<!ENTITY ccedil "&#231;" ><!-- latin small letter c
                                with cedilla,
                                U+00E7 ISOlat1 ->
<!ENTITY egrave "&#232;" ><!-- latin small letter e with grave,
                                U+00E8 ISOlat1 ->
<!ENTITY eacute "&#233;" ><!-- latin small letter e with acute,
                                U+00E9 ISOlat1 ->
<!ENTITY ecirc "&#234;" ><!-- latin small letter e
                                with circumflex,
                                U+00EA ISOlat1 ->
<!ENTITY euml "&#235;" ><!-- latin small letter e
                                with diaeresis,
                                U+00EB ISOlat1 ->
<!ENTITY igrave "&#236;" ><!-- latin small letter i with grave,
                                U+00EC ISOlat1 ->
<!ENTITY iacute "&#237;" ><!-- latin small letter i with acute,
                                U+00ED ISOlat1 ->
<!ENTITY icirc "&#238;" ><!-- latin small letter i
                                with circumflex,
                                U+00EE ISOlat1 ->
<!ENTITY iuml "&#239;" ><!-- latin small letter I
                                with diaeresis,
                                U+00EF ISOlat1 ->
<!ENTITY eth "&#240;" ><!-- latin small letter eth,
                                U+00F0 ISOlat1 ->
<!ENTITY ntilde "&#241;" ><!-- latin small letter n with tilde,
                                U+00F1 ISOlat1 ->
<!ENTITY ograve "&#242;" ><!-- latin small letter o with grave,
                                U+00F2 ISOlat1 ->
<!ENTITY oacute "&#243;" ><!-- latin small letter o with acute,
                                U+00F3 ISOlat1 ->
<!ENTITY ocirc "&#244;" ><!-- latin small letter o
                                with circumflex,
                                U+00F4 ISOlat1 ->
<!ENTITY otilde "&#245;" ><!-- latin small letter o with tilde,
                                U+00F5 ISOlat1 ->
<!ENTITY ouml "&#246;" ><!-- latin small letter o
                                with diaeresis,
                                U+00F6 ISOlat1 ->
<!ENTITY divide "&#247;" ><!-- division sign,
                                U+00F7 ISOnum ->
<!ENTITY slash "&#248;" ><!-- latin small letter o with stroke,
                                = latin small letter o slash,
                                U+00F8 ISOlat1 ->
<!ENTITY ugrave "&#249;" ><!-- latin small letter u with grave,
                                U+00F9 ISOlat1 ->
<!ENTITY uacute "&#250;" ><!-- latin small letter u with acute,
                                U+00FA ISOlat1 ->
<!ENTITY ucirc "&#251;" ><!-- latin small letter u
                                with circumflex,

```

Continued

Listing 20-30 (continued)

```

                                U+00FB ISOlat1 →
<!ENTITY uuml "&#252;" ><!-- latin small letter u
                                with diaeresis,
                                U+00FC ISOlat1 →
<!ENTITY yacute "&#253;" ><!-- latin small letter y with acute,
                                U+00FD ISOlat1 →
<!ENTITY thorn "&#254;" ><!-- latin small letter thorn with,
                                U+00FE ISOlat1 →
<!ENTITY yuml "&#255;" ><!-- latin small letter y
                                with diaeresis,
                                U+00FF ISOlat1 →

```

The XHTML Special Character Entities

XHTML1-special.ent, shown in Listing 20-31, defines the general entities for an assortment of characters not in Latin-1, but present in Unicode.

Listing 20-31: XHTML1-special.ent: the XHTML definitions for a few character entities that don't really fit anywhere else

```

<!--
XML-compatible ISO Special Character Entity Set for XHTML 1.0

Typical invocation:

    <!ENTITY % XHTML1-special
        PUBLIC "-//W3C//ENTITIES Special//EN//XML"
        "XHTML1-special.ent">
    %XHTML1-special;

Revision: @(#)XHTML1-special.ent 1.13 99/04/01 SMI

Portions (C) International Organization for
Standardization 1986: Permission to copy in any form is
granted for use with conforming SGML systems and
applications as defined in ISO 8879, provided this notice
is included in all copies.
-->

<!-- Relevant ISO entity set is given unless names are newly
introduced. New names (i.e., not in ISO 8879 list) do not
clash with any existing ISO 8879 entity names. ISO 10646
character numbers are given for each character, in hex.

```

CDATA values are decimal conversions of the ISO 10646 values and refer to the document character set. Names are Unicode 2.0 names.

→

```

<!-- C0 Controls and Basic Latin -->
<!ENTITY quot "&#34;"> <!-- quotation mark = APL quote,
                        U+0022 ISOnum -->
<!ENTITY amp  "&#38;"> <!-- ampersand, U+0026 ISOnum -->
<!ENTITY lt   "&#60;"> <!-- less-than sign, U+003C ISOnum-->
<!ENTITY gt   "&#62;"> <!-- greater-than sign, U+003E ISOnum-->

<!-- Latin Extended-A -->
<!ENTITY OElig "&#338;"> <!-- latin capital ligature OE,
                        U+0152 ISOlat2 -->
<!ENTITY oelig "&#339;"> <!-- latin small ligature oe,
                        U+0153 ISOlat2 -->
<!-- ligature is a misnomer, this is a separate character
      in some languages -->
<!ENTITY Scaron "&#352;"> <!-- latin capital letter S
                        with caron,
                        U+0160 ISOlat2 -->
<!ENTITY scaron "&#353;"> <!-- latin small letter s
                        with caron,
                        U+0161 ISOlat2 -->
<!ENTITY Yuml   "&#376;"> <!-- latin capital letter Y
                        with diaeresis,
                        U+0178 ISOlat2 -->

<!-- Spacing Modifier Letters -->
<!ENTITY circ  "&#710;"> <!-- modifier letter
                        circumflex accent,
                        U+02C6 ISOpub -->
<!ENTITY tilde "&#732;"> <!-- small tilde, U+02DC ISodia -->

<!-- General Punctuation -->
<!ENTITY ensp  "&#8194;"> <!-- en space, U+2002 ISOpub -->
<!ENTITY emsp  "&#8195;"> <!-- em space, U+2003 ISOpub -->
<!ENTITY thinsp "&#8201;"> <!-- thin space, U+2009 ISOpub -->
<!ENTITY zwnj  "&#8204;"> <!-- zero width non-joiner,
                        U+200C NEW RFC 2070 -->
<!ENTITY zwj   "&#8205;"> <!-- zero width joiner,
                        U+200D NEW RFC 2070 -->
<!ENTITY lrm   "&#8206;"> <!-- left-to-right mark,
                        U+200E NEW RFC 2070 -->
<!ENTITY rlm   "&#8207;"> <!-- right-to-left mark,
                        U+200F NEW RFC 2070 -->
<!ENTITY ndash "&#8211;"> <!-- en dash, U+2013 ISOpub -->
<!ENTITY mdash "&#8212;"> <!-- em dash, U+2014 ISOpub -->
<!ENTITY lsquo "&#8216;"> <!-- left single quotation mark,
                        U+2018 ISOnum -->

```

Continued

Listing 20-31 (continued)

```

<!ENTITY rsquo    "’"> <!-- right single quotation mark,
                                U+2019 ISOnum ->
<!ENTITY sbquo    "‚"> <!-- single low-9 quotation mark,
                                U+201A NEW ->
<!ENTITY ldquo    "“"> <!-- left double quotation mark,
                                U+201C ISOnum ->
<!ENTITY rdquo    "”"> <!-- right double quotation mark,
                                U+201D ISOnum ->
<!ENTITY bdquo    "„"> <!-- double low-9 quotation mark,
                                U+201E NEW ->
<!ENTITY dagger    "†"> <!-- dagger, U+2020 ISOpub ->
<!ENTITY Dagger    "‡"> <!-- double dagger,
                                U+2021 ISOpub ->
<!ENTITY permil    "‰"> <!-- per mille sign,
                                U+2030 ISOtech ->
<!ENTITY lsaquo    "‹"> <!-- single left-pointing angle
                                quotation mark,
                                U+2039 ISO proposed ->
<!-- lsaquo is proposed but not yet ISO standardized ->
<!ENTITY rsaquo    "›"> <!-- single right-pointing
                                angle quotation mark,
                                U+203A ISO proposed ->
<!-- rsaquo is proposed but not yet ISO standardized ->
<!ENTITY euro      "€"> <!-- euro sign, U+20AC NEW ->

```

The XHTML Symbol Entities

XHTML1-symbol.ent, shown in Listing 20-32, defines the general entities for the Greek alphabet and various mathematical symbols like the integral and square root signs.

Listing 20-32: XHTML1-symbol.ent: the Voyager entity set for mathematical symbols, including the Greek alphabet

```

<!-- XML-compatible ISO Mathematical, Greek and Symbolic
      Character Entity Set for XHTML 1.0

```

Typical invocation:

```

<!ENTITY % XHTML1-symbol
      PUBLIC "-//W3C//ENTITIES Symbols//EN//XML"
      "XHTML1-symbol.ent">
%XHTML1-symbol;

```

```

Revision:  @(#)XHTML1-symbol.ent 1.13 99/04/01 SMI

Portions (C) International Organization for
Standardization 1986: Permission to copy in any form is
granted for use with conforming SGML systems and
applications as defined in ISO 8879, provided this notice
is included in all copies.
->

<!-- Relevant ISO entity set is given unless names are newly
introduced. New names (i.e., not in ISO 8879 list) do not
clash with any existing ISO 8879 entity names. ISO 10646
character numbers are given for each character, in hex.
CDATA values are decimal conversions of the ISO 10646
values and refer to the document character set. Names are
Unicode 2.0 names.
->

<!-- Latin Extended-B ->
<!ENTITY fnof      "&#402;"> <!-- latin small f with hook
                        = function
                        = florin, U+0192 IS0tech>

<!-- Greek ->
<!ENTITY Alpha    "&#913;" ><!-- greek capital letter alpha,
                        U+0391 ->
<!ENTITY Beta     "&#914;" ><!-- greek capital letter beta,
                        U+0392 ->
<!ENTITY Gamma    "&#915;" ><!-- greek capital letter gamma,
                        U+0393 ISOgrk3 ->
<!ENTITY Delta    "&#916;" ><!-- greek capital letter delta,
                        U+0394 ISOgrk3 ->
<!ENTITY Epsilon  "&#917;" ><!-- greek capital letter epsilon,
                        U+0395 ->
<!ENTITY Zeta     "&#918;" ><!-- greek capital letter zeta,
                        U+0396 ->
<!ENTITY Eta      "&#919;" ><!-- greek capital letter eta,
                        U+0397 ->
<!ENTITY Theta    "&#920;" ><!-- greek capital letter theta,
                        U+0398 ISOgrk3 ->
<!ENTITY Iota     "&#921;" ><!-- greek capital letter iota,
                        U+0399 ->
<!ENTITY Kappa    "&#922;" ><!-- greek capital letter kappa,
                        U+039A ->
<!ENTITY Lambda   "&#923;" ><!-- greek capital letter lambda,
                        U+039B ISOgrk3 ->
<!ENTITY Mu       "&#924;" ><!-- greek capital letter mu,
                        U+039C ->
<!ENTITY Nu       "&#925;" ><!-- greek capital letter nu,
                        U+039D ->

```

Continued

Listing 20-32 (continued)

```

<!ENTITY Xi      "&#926;" ><!-- greek capital letter xi,
                    U+039E ISOgrk3 ->
<!ENTITY Omicron "&#927;" ><!-- greek capital letter omicron,
                    U+039F ->
<!ENTITY Pi      "&#928;" ><!-- greek capital letter pi,
                    U+03A0 ISOgrk3 ->
<!ENTITY Rho     "&#929;" ><!-- greek capital letter rho,
                    U+03A1 ->
<!-- there is no Sigmaf, and no U+03A2 character either ->
<!ENTITY Sigma   "&#931;" ><!-- greek capital letter sigma,
                    U+03A3 ISOgrk3 ->
<!ENTITY Tau     "&#932;" ><!-- greek capital letter tau,
                    U+03A4 ->
<!ENTITY Upsilon "&#933;" ><!-- greek capital letter upsilon,
                    U+03A5 ISOgrk3 ->
<!ENTITY Phi     "&#934;" ><!-- greek capital letter phi,
                    U+03A6 ISOgrk3 ->
<!ENTITY Chi     "&#935;" ><!-- greek capital letter chi,
                    U+03A7 ->
<!ENTITY Psi     "&#936;" ><!-- greek capital letter psi,
                    U+03A8 ISOgrk3 ->
<!ENTITY Omega   "&#937;" ><!-- greek capital letter omega,
                    U+03A9 ISOgrk3 ->
<!ENTITY alpha   "&#945;" ><!-- greek small letter alpha,
                    U+03B1 ISOgrk3 ->
<!ENTITY beta    "&#946;" ><!-- greek small letter beta,
                    U+03B2 ISOgrk3 ->
<!ENTITY gamma   "&#947;" ><!-- greek small letter gamma,
                    U+03B3 ISOgrk3 ->
<!ENTITY delta   "&#948;" ><!-- greek small letter delta,
                    U+03B4 ISOgrk3 ->
<!ENTITY epsilon "&#949;" ><!-- greek small letter epsilon,
                    U+03B5 ISOgrk3 ->
<!ENTITY zeta    "&#950;" ><!-- greek small letter zeta,
                    U+03B6 ISOgrk3 ->
<!ENTITY eta     "&#951;" ><!-- greek small letter eta, U+03B7
                    ISOgrk3 ->
<!ENTITY theta   "&#952;" ><!-- greek small letter theta,
                    U+03B8 ISOgrk3 ->
<!ENTITY iota    "&#953;" ><!-- greek small letter iota,
                    U+03B9 ISOgrk3 ->
<!ENTITY kappa   "&#954;" ><!-- greek small letter kappa,
                    U+03BA ISOgrk3 ->
<!ENTITY lambda  "&#955;" ><!-- greek small letter lambda,
                    U+03BB ISOgrk3 ->
<!ENTITY mu      "&#956;" ><!-- greek small letter mu, U+03BC
                    ISOgrk3 ->
<!ENTITY nu      "&#957;" ><!-- greek small letter nu, U+03BD
                    ISOgrk3 ->
<!ENTITY xi      "&#958;" ><!-- greek small letter xi, U+03BE
                    ISOgrk3 ->

```

```

<!ENTITY omicron "&#959;" ><!-- greek small letter omicron,
U+03BF NEW ->
<!ENTITY pi "&#960;" ><!-- greek small letter pi,
U+03C0 ISOgrk3 ->
<!ENTITY rho "&#961;" ><!-- greek small letter rho,
U+03C1 ISOgrk3 ->
<!ENTITY sigmaf "&#962;" ><!-- greek small letter final
sigma, U+03C2 ISOgrk3 ->
<!ENTITY sigma "&#963;" ><!-- greek small letter sigma,
U+03C3 ISOgrk3 ->
<!ENTITY tau "&#964;" ><!-- greek small letter tau,
U+03C4 ISOgrk3 ->
<!ENTITY upsilon "&#965;" ><!-- greek small letter upsilon,
U+03C5 ISOgrk3 ->
<!ENTITY phi "&#966;" ><!-- greek small letter phi,
U+03C6 ISOgrk3 ->
<!ENTITY chi "&#967;" ><!-- greek small letter chi,
U+03C7 ISOgrk3 ->
<!ENTITY psi "&#968;" ><!-- greek small letter psi,
U+03C8 ISOgrk3 ->
<!ENTITY omega "&#969;" ><!-- greek small letter omega,
U+03C9 ISOgrk3 ->
<!ENTITY thetasym "&#977;" ><!-- greek small letter theta
symbol, U+03D1 NEW ->
<!ENTITY upsih "&#978;" ><!-- greek upsilon with hook
symbol, U+03D2 NEW ->
<!ENTITY piv "&#982;" ><!-- greek pi symbol,
U+03D6 ISOgrk3 ->

<!-- General Punctuation ->
<!ENTITY bull "&#8226;" ><!-- bullet = black small circle,
U+2022 ISOpub ->
<!-- bullet is NOT the same as bullet operator, U+2219 ->
<!ENTITY hellip "&#8230;" ><!-- horizontal ellipsis
= three dot leader, U+2026 ISOpub ->
<!ENTITY prime "&#8242;" ><!-- prime = minutes = feet,
U+2032 ISOtech ->
<!ENTITY Prime "&#8243;" ><!-- double prime = seconds
= inches, U+2033 ISOtech ->
<!ENTITY oline "&#8254;" ><!-- overline = spacing overscore,
U+203E NEW ->
<!ENTITY frasl "&#8260;" ><!-- fraction slash, U+2044 NEW->

<!-- Letterlike Symbols ->
<!ENTITY weierp "&#8472;" ><!-- script capital P = power set
= Weierstrass p, U+2118 ISOamso ->
<!ENTITY image "&#8465;" ><!-- blackletter capital I
= imaginary part, U+2111 ISOamso ->
<!ENTITY real "&#8476;" ><!-- blackletter capital R
= real part symbol, U+211C ISOamso ->
<!ENTITY trade "&#8482;" ><!-- trade mark sign,
U+2122 ISOnum ->

```

Continued

Listing 20-32 (continued)

```

<!ENTITY alefsym "ℵ" ><!-- alef symbol
           = first transfinite cardinal, U+2135 NEW ->
<!-- alef symbol is NOT the same as hebrew letter alef,
           U+05D0 although the same glyph could be used to depict
           both characters ->

<!-- Arrows ->
<!ENTITY larr    "←" ><!-- leftwards arrow,
           U+2190 ISOnum ->
<!ENTITY uarr    "↑" ><!-- upwards arrow, U+2191 ISOnum->
<!ENTITY rarr    "→" ><!-- rightwards arrow,
           U+2192 ISOnum ->
<!ENTITY darr    "↓" ><!-- downwards arrow,
           U+2193 ISOnum ->
<!ENTITY harr    "↔" ><!-- left right arrow,
           U+2194 ISOamsa ->
<!ENTITY crarr   "↵" ><!-- downwards arrow with corner
           leftwards = carriage return, U+21B5 NEW ->
<!ENTITY lArr    "⇐" ><!-- leftwards double arrow,
           U+21D0 ISOtech ->
<!-- Unicode does not say that lArr is the same as the
           'is implied by' arrow but also does not have any other
           character for that function. So ? lArr can
           be used for 'is implied by' as ISOtech suggests ->
<!ENTITY uArr    "⇑" ><!-- upwards double arrow,
           U+21D1 ISOamsa ->
<!ENTITY rArr    "⇒" ><!-- rightwards double arrow,
           U+21D2 ISOtech ->
<!-- Unicode does not say this is the 'implies' character
           but does not have another character with this function
           so ? rArr can be used for 'implies' as ISOtech suggests ->
<!ENTITY dArr    "⇓" ><!-- downwards double arrow,
           U+21D3 ISOamsa ->
<!ENTITY hArr    "⇔" ><!-- left right double arrow,
           U+21D4 ISOamsa ->

<!-- Mathematical Operators ->
<!ENTITY forall  "∀" ><!-- for all, U+2200 ISOtech ->
<!ENTITY part    "∂" ><!-- partial differential,
           U+2202 ISOtech ->
<!ENTITY exist   "∃" ><!--there exists, U+2203 ISOtech->
<!ENTITY empty   "∅" ><!-- empty set = null set
           = diameter, U+2205 ISOamso ->
<!ENTITY nabla   "∇" ><!-- nabla = backward difference,
           U+2207 ISOtech ->
<!ENTITY isin    "∈" ><!-- element of, U+2208 ISOtech->
<!ENTITY notin   "∉" ><!-- not an element of,
           U+2209 ISOtech ->
<!ENTITY ni      "∋" ><!-- contains as member,
           U+220B ISOtech ->
<!-- should there be a more memorable name than 'ni'? ->

```

```

<!ENTITY prod      "&#8719;" ><!-- n-ary product = product sign,
                        U+220F ISOamsb ->
<!-- prod is NOT the same character as U+03A0 'greek capital
      letter pi' though the same glyph might be used for both->
<!ENTITY sum       "&#8721;" ><!-- n-ary sumation,
                        U+2211 ISOamsb ->
<!-- sum is NOT the same character as U+03A3
      'greek capital letter sigma' though the same glyph
      might be used for both ->
<!ENTITY minus     "&#8722;" ><!-- minus sign, U+2212 ISOtech->
<!ENTITY lowast    "&#8727;" ><!-- asterisk operator,
                        U+2217 ISOtech ->
<!ENTITY radic     "&#8730;" ><!-- square root = radical sign,
                        U+221A ISOtech ->
<!ENTITY prop      "&#8733;" ><!-- proportional to,
                        U+221D ISOtech ->
<!ENTITY infin     "&#8734;" ><!-- infinity, U+221E ISOtech ->
<!ENTITY ang       "&#8736;" ><!-- angle, U+2220 ISOamso ->
<!ENTITY and       "&#8743;" ><!-- logical and = wedge,
                        U+2227 ISOtech ->
<!ENTITY or        "&#8744;" ><!-- logical or = vee,
                        U+2228 ISOtech ->
<!ENTITY cap       "&#8745;" ><!-- intersection = cap,
                        U+2229 ISOtech ->
<!ENTITY cup       "&#8746;" ><!-- union = cup, U+222A ISOtech->
<!ENTITY int       "&#8747;" ><!-- integral, U+222B ISOtech ->
<!ENTITY there4    "&#8756;" ><!-- therefore, U+2234 ISOtech ->
<!ENTITY sim       "&#8764;" ><!-- tilde operator
                        = varies with = similar to, U+223C ISOtech ->
<!-- tilde operator is NOT the same character as the tilde,
      U+007E, although the same glyph might be used to
      represent both ->
<!ENTITY cong      "&#8773;" ><!-- approximately equal to, U+2245
ISOtech ->
<!ENTITY asymp     "&#8776;" ><!-- almost equal to
                        = asymptotic to, U+2248 ISOamsr ->
<!ENTITY ne        "&#8800;" ><!-- not equal to,
                        U+2260 ISOtech ->
<!ENTITY equiv     "&#8801;" ><!-- identical to,
                        U+2261 ISOtech ->
<!ENTITY le        "&#8804;" ><!-- less-than or equal to,
                        U+2264 ISOtech ->
<!ENTITY ge        "&#8805;" ><!-- greater-than or equal to,
                        U+2265 ISOtech ->
<!ENTITY sub       "&#8834;" ><!-- subset of, U+2282 ISOtech ->
<!ENTITY sup       "&#8835;" ><!-- superset of, U+2283 ISOtech->
<!-- note that nsup, 'not a superset of, U+2283' is not covered
      by the Symbol font encoding and is not included. Should it
      be, for symmetry? It is in ISOamsn ->
<!ENTITY nsub      "&#8836;" ><!-- not a subset of,
                        U+2284 ISOamsn ->

```

Continued

Listing 20-32 (continued)

```

<!ENTITY sube      "&#8838;" ><!-- subset of or equal to,
                    U+2286 ISOtech ->
<!ENTITY supe      "&#8839;" ><!-- superset of or equal to,
                    U+2287 ISOtech ->
<!ENTITY oplus      "&#8853;" ><!-- circled plus = direct sum,
                    U+2295 ISOamsb ->
<!ENTITY otimes     "&#8855;" ><!-- circled times
                    = vector product, U+2297 ISOamsb ->
<!ENTITY perp       "&#8869;" ><!-- up tack = orthogonal to
                    = perpendicular, U+22A5 ISOtech ->
<!ENTITY sdot       "&#8901;" ><!-- dot operator,
                    U+22C5 ISOamsb ->
<!-- dot operator is NOT the same character as
      U+00B7 middle dot ->

<!-- Miscellaneous Technical ->
<!ENTITY lceil      "&#8968;" ><!-- left ceiling = apl upstile,
                    U+2308 ISOamsc ->
<!ENTITY rceil      "&#8969;" ><!-- right ceiling,
                    U+2309 ISOamsc ->
<!ENTITY lfloor     "&#8970;" ><!-- left floor = apl downstile,
                    U+230A ISOamsc ->
<!ENTITY rfloor     "&#8971;" ><!-- right floor,
                    U+230B ISOamsc ->
<!ENTITY lang       "&#9001;" ><!-- left-pointing angle bracket
                    = bra, U+2329 ISOtech ->
<!-- lang is NOT the same character as U+003C 'less than'
      or U+2039 'single left-pointing angle quotation mark' ->
<!ENTITY rang       "&#9002;" ><!-- right-pointing angle bracket
                    = ket, U+232A ISOtech ->
<!-- rang is NOT the same character as U+003E 'greater than'
      or U+203A 'single right-pointing angle quotation mark' ->

<!-- Geometric Shapes ->
<!ENTITY loz        "&#9674;" ><!-- lozenge, U+25CA ISOpub ->

<!-- Miscellaneous Symbols ->
<!ENTITY spades     "&#9824;" ><!-- black spade suit,
                    U+2660 ISOpub ->
<!-- black here seems to mean filled as opposed to hollow ->
<!ENTITY clubs      "&#9827;" ><!-- black club suit = shamrock,
                    U+2663 ISOpub ->
<!ENTITY hearts     "&#9829;" ><!-- black heart suit = valentine,
                    U+2665 ISOpub ->
<!ENTITY diams      "&#9830;" ><!-- black diamond suit,
                    U+2666 ISOpub ->

```

Simplified Subset DTDs

Not all HTML-based systems need every piece of HTML. Depending on your needs, you may well be able to omit forms, applets, images, image maps, and other advanced, interactive features of HTML. For instance, returning to the baseball examples of Part I, if you were to give each `PLAYER` a `BIO` element, you could use simple HTML to include basic text with each player.

The key modules that you'll probably want to include in any application you design using XHTML are:

- ♦ XHTML1-attrs.mod
- ♦ XHTML1-blkphras.mod
- ♦ XHTML1-blkpres.mod
- ♦ XHTML1-blkstruct.mod
- ♦ XHTML1-charent.mod
- ♦ XHTML1-inlphras.mod
- ♦ XHTML1-inlpres.mod
- ♦ XHTML1-inlstruct.mod
- ♦ XHTML1-model.mod
- ♦ XHTML1-names.mod

In addition, it's easy to mix in other modules to this basic set. For instance, `XHTML1-image` for images or `XHTML1-linking` for hypertext. While you can link these into your own DTDs using external parameter entity references (as you'll see an example of in Chapter 23), the simplest way to choose the parts you do and don't want is to copy either the transitional or strict DTD and `IGNORE` the parts you don't want. Listing 20-33 is a copy of the strict DTD (Listing 20-1) in which only the modules listed above are included:

Listing 20-33: A core DTD that supports basic HTML

```
<!-- ..... ->
<!-- Basic HTML for Player BIOs, based on XHTML 1.0 strict ->
<!-- file: XHTML1-bb.dtd
->

<!-- This derived from
      XHTML 1.0, an XML reformulation of HTML 4.0.
```

Copyright 1998-1999 World Wide Web Consortium

Continued

Listing 20-33 (continued)

(Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved.

Permission to use, copy, modify and distribute the XHTML 1.0 DTD and its accompanying documentation for any purpose and without fee is hereby granted in perpetuity, provided that the above copyright notice and this paragraph appear in all copies. The copyright holders make no representation about the suitability of the DTD for any purpose.

It is provided "as is" without expressed or implied warranty.

Original Author: Murray M. Altheim <altheim@eng.sun.com>
Original Revision: @(#)XHTML1-s.dtd 1.14 99/04/01 SMI

The DTD is an XML variant based on the
W3C HTML 4.0 DTD:

Draft: \$Date: 1999/04/02 14:27:27 \$

Authors: Dave Raggett <dsr@w3.org>
Arnaud Le Hors <lehors@w3.org>
Ian Jacobs <ij@w3.org>

→

<!-- The version attribute has historically been a container for the DTD's public identifier (an FPI), but is unused in Strict: →

```
<!ENTITY % HTML.version "" >
<!ENTITY % Version.attrib "" >
```

<!-- The xmlns attribute on <html> identifies the default namespace to namespace-aware applications: →

```
<!ENTITY % XHTML.ns "http://www.w3.org/TR/1999/REC-html-in-xml" >
```

<!-- reserved for future use with document profiles →

```
<!ENTITY % XHTML.profile "" >
```

<!-- used to ignore Transitional features within modules →

```
<!ENTITY % XHTML.Transitional "IGNORE" >
```

<!-- XHTML Base Architecture Module (optional) →

```
<!ENTITY % XHTML1-arch.module "IGNORE" >
<![%XHTML1-arch.module;[
<!ENTITY % XHTML1-arch.mod
PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Base Architecture//EN"
```

```

        "XHTML1-arch.mod" >
%XHTML1-arch.mod;
]]>

<!-- Common Names Module ..... ->
<!ENTITY % XHTML1-names.module "INCLUDE" >
<![%XHTML1-names.module;[
<!ENTITY % XHTML1-names.mod
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Common Names//EN"
        "XHTML1-names.mod" >
%XHTML1-names.mod;
]]>

<!-- Character Entities Module ..... ->
<!ENTITY % XHTML1-charent.module "INCLUDE" >
<![%XHTML1-charent.module;[
<!ENTITY % XHTML1-charent.mod
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Character Entities//EN"
        "XHTML1-charent.mod" >
%XHTML1-charent.mod;
]]>

<!-- Intrinsic Events Module ..... ->
<!ENTITY % XHTML1-events.module "IGNORE" >
<![%XHTML1-events.module;[
<!ENTITY % XHTML1-events.mod
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Intrinsic Events//EN"
        "XHTML1-events.mod" >
%XHTML1-events.mod;
]]>

<!-- Common Attributes Module ..... ->
<!ENTITY % XHTML1-attribs.module "INCLUDE" >
<![%XHTML1-attribs.module;[
<!ENTITY % align "" >
<!ENTITY % XHTML1-attribs.mod
    PUBLIC "-//W3C//ENTITIES XHTML 1.0 Common Attributes//EN"
        "XHTML1-attribs.mod" >
%XHTML1-attribs.mod;
]]>

<!-- Document Model Module ..... ->
<!ENTITY % XHTML1-model.module "INCLUDE" >
<![%XHTML1-model.module;[
<!ENTITY % XHTML1-model.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Document Model//EN"
        "XHTML1-model.mod" >
%XHTML1-model.mod;
]]>

<!-- Inline Structural Module ..... ->
<!ENTITY % XHTML1-inlstruct.module "INCLUDE" >

```

Continued

Listing 20-33 (continued)

```

<![%XHTML1-inlstruct.module;[
<!ENTITY % XHTML1-inlstruct.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Structural//EN"
        "XHTML1-inlstruct.mod" >
%XHTML1-inlstruct.mod;
]]>

<!-- Inline Presentational Module ..... ->
<!ENTITY % XHTML1-inlpres.module "INCLUDE" >
<![%XHTML1-inlpres.module;[
<!ENTITY % XHTML1-inlpres.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Presentational//EN"
        "XHTML1-inlpres.mod" >
%XHTML1-inlpres.mod;
]]>

<!-- Inline Phrasal Module ..... ->
<!ENTITY % XHTML1-inlphras.module "INCLUDE" >
<![%XHTML1-inlphras.module;[
<!ENTITY % XHTML1-inlphras.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Inline Phrasal//EN"
        "XHTML1-inlphras.mod" >
%XHTML1-inlphras.mod;
]]>

<!-- Block Structural Module ..... ->
<!ENTITY % XHTML1-blkstruct.module "INCLUDE" >
<![%XHTML1-blkstruct.module;[
<!ENTITY % XHTML1-blkstruct.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Structural//EN"
        "XHTML1-blkstruct.mod" >
%XHTML1-blkstruct.mod;
]]>

<!-- Block Presentational Module ..... ->
<!ENTITY % XHTML1-blkpres.module "INCLUDE" >
<![%XHTML1-blkpres.module;[
<!ENTITY % XHTML1-blkpres.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block
Presentational//EN"
        "XHTML1-blkpres.mod" >
%XHTML1-blkpres.mod;
]]>

<!-- Block Phrasal Module ..... ->
<!ENTITY % XHTML1-blkphras.module "INCLUDE" >
<![%XHTML1-blkphras.module;[
<!ENTITY % XHTML1-blkphras.mod

```

```

        PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Block Phrasal//EN"
            "XHTML1-blkphras.mod" >
%XHTML1-blkphras.mod;
]]>

<!-- Scripting Module ..... ->
<!ENTITY % XHTML1-script.module "IGNORE" >
<![%XHTML1-script.module;[
<!ENTITY % XHTML1-script.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Scripting//EN"
        "XHTML1-script.mod" >
%XHTML1-script.mod;
]]>

<!-- Stylesheets Module ..... ->
<!ENTITY % XHTML1-style.module "IGNORE" >
<![%XHTML1-style.module;[
<!ENTITY % XHTML1-style.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Stylesheets//EN"
        "XHTML1-style.mod" >
%XHTML1-style.mod;
]]>

<!-- Image Module ..... ->
<!ENTITY % XHTML1-image.module "IGNORE" >
<![%XHTML1-image.module;[
<!ENTITY % XHTML1-image.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Images//EN"
        "XHTML1-image.mod" >
%XHTML1-image.mod;
]]>

<!-- Frames Module ..... ->
<!ENTITY % XHTML1-frames.module "IGNORE" >
<![%XHTML1-frames.module;[
<!ENTITY % XHTML1-frames.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Frames//EN"
        "XHTML1-frames.mod" >
%XHTML1-frames.mod;
]]>

<!-- Linking Module ..... ->
<!ENTITY % XHTML1-linking.module "IGNORE" >
<![%XHTML1-linking.module;[
<!ENTITY % XHTML1-linking.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Linking//EN"
        "XHTML1-linking.mod" >
%XHTML1-linking.mod;
]]>

```

Continued

Listing 20-33 (continued)

```

<!-- Client-side Image Map Module ..... ->
<!ENTITY % XHTML1-csismap.module "IGNORE" >
<![%XHTML1-csismap.module;[
<!ENTITY % XHTML1-csismap.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Client-side Image Map//EN"
        "XHTML1-csismap.mod" >
%XHTML1-csismap.mod;
]]>

<!-- Object Element Module ..... ->
<!ENTITY % XHTML1-object.module "IGNORE" >
<![%XHTML1-object.module;[
<!ENTITY % XHTML1-object.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Object Element//EN"
        "XHTML1-object.mod" >
%XHTML1-object.mod;
]]>

<!-- Lists Module ..... ->
<!ENTITY % XHTML1-list.module "IGNORE" >
<![%XHTML1-list.module;[
<!ENTITY % XHTML1-list.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Lists//EN"
        "XHTML1-list.mod" >
%XHTML1-list.mod;
]]>

<!-- Forms Module ..... ->
<!ENTITY % XHTML1-form.module "IGNORE" >
<![%XHTML1-form.module;[
<!ENTITY % XHTML1-form.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Forms//EN"
        "XHTML1-form.mod" >
%XHTML1-form.mod;
]]>

<!-- Tables Module ..... ->
<!ENTITY % XHTML1-table.module "IGNORE" >
<![%XHTML1-table.module;[
<!ENTITY % XHTML1-table.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Tables//EN"
        "XHTML1-table.mod" >
%XHTML1-table.mod;
]]>

<!-- Document Metainformation Module ..... ->
<!ENTITY % XHTML1-meta.module "IGNORE" >
<![%XHTML1-meta.module;[

```

```

<!ENTITY % XHTML1-meta.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Metainformation//EN"
        "XHTML1-meta.mod" >
%XHTML1-meta.mod;
]]>

<!-- Document Structure Module ..... ->
<!ENTITY % XHTML1-struct.module "IGNORE" >
<![%XHTML1-struct.module;[
<!ENTITY % XHTML1-struct.mod
    PUBLIC "-//W3C//ELEMENTS XHTML 1.0 Document Structure//EN"
        "XHTML1-struct.mod" >
%XHTML1-struct.mod;
]]>

<!-- end of XHTML 1.0 Strict DTD ..... ->
<!-- ..... ->

```

Aside from some changes to the comments at the top to indicate that this is a derived version of the XHTML strict DTD, the only changes are the replacement of INCLUDE by IGNORE in several parameter entity references like XHTML1-struct.module.

It would also be possible to simply delete the unnecessary sections completely, rather than simply ignoring them. However, this approach makes it very easy to include them quickly if a need for them is discovered in the future.

You can't call the resulting application HTML, but it does provide a neat way to add basic hypertext structure to a more domain-specific DTD without going overboard and pulling in the full multimedia smorgasbord that is HTML 4.0.

For example, by adding Listing 20-33 to the DTD for baseball players from Chapter 10, I could give each player a BIOGRAPHY element that contains basic HTML. The declarations would look like this:

```

<!ENTITY % XHTML1-bb.dtd SYSTEM "XHTML1-bb.dtd">

%XHTML1-bb.dtd;
<!ENTITY % BIOGRAPHY.content "( #PCDATA | %Flow.mix; )*" >
<ELEMENT BIOGRAPHY %BIOGRAPHY.content;>

```

This says that a BIOGRAPHY can contain anything an HTML block can contain as defined by the XHTML modules used here. If you prefer, you can use any of the other elements or content model entity references from the XHTML modules.

Copyright Notices in DTDs

If you're designing a DTD solely for your use on your own Web site or for printed documentation within a single company, feel free to place any copyright notice you want on it. However, if you're designing a DTD for an entire industry or area of study, please consider any copyright notice very carefully. A simple, ordinary copyright notice like "Copyright 1999 Elliotte Rusty Harold" immediately makes the DTD unusable for many people because by default it means the DTD can't be copied onto a different Web server or into a new document without explicit permission. While many people and companies will simply ignore these restrictions (which the authors never intended anyway), I don't think many people will be comfortable relying on this in our overly litigious world.

The whole point of XML is to allow broad, standardized documents. To this end, any markup language that's created, whether described in a DTD, a DCD, a DDML DocumentDef, or something else, must explicitly allow itself to be reused and reprinted without prior permission. My preference is that these DTDs be placed in the public domain, because it's simplest and easiest to explain to lawyers. Open source works well too. Even a copyright statement that allows reuse but not modification is adequate for many needs.

Therefore, I implore you to think very carefully about any copyright you place on a DTD. Ask yourself, "What does this really say? What do I want people to do with this DTD? Does this statement allow them to do that?" There's very little to be gained by writing a DTD you hope an industry will adopt, if you unintentionally prohibit the industry from adopting it.

(Although this book as a whole and its prose text is copyrighted, I am explicitly placing the code examples I've written in the public domain. Please feel free to use any fragment of code or an entire DTD in any way that you like, with or without credit.)

Techniques to Imitate

Pablo Picasso is often quoted as saying, "Good artists copy. Great artists steal." As you've already seen, part of the reason the XHTML DTD is so modular — broken up into so many parts — is precisely so that you can steal from it. If you need basic hypertext formatting as part of an XML application you're developing, you really don't need to invent your own. You can simply import the necessary modules. This has the added advantage that document authors who have to use your XML application are likely already familiar with this markup from HTML. Nonetheless, let's go ahead and look at some techniques you can borrow from the XHTML DTD for your own DTDs without out-and-out stealing the DTDs themselves.

Comments

The XHTML DTDs are profusely commented. Every single file has a comment that gives a title, the relevant copyright notice, and an abstract of what's in the file, before there's even one single declaration. Every section of the file is separated off by a new comment that specifies the purpose of the section. And almost every

declaration features a comment discussing what that declaration means. This all makes the file *much* easier to read and understand.

This still isn't perfect, however. Many of the attribute declarations are not sufficiently commented. For example, consider this declaration from XHTML1-applet.mod:

```
<!ATTLIST applet
  %Core.attrib;
  codebase      %URI;                #IMPLIED
  archive       CDATA                 #IMPLIED
  code          CDATA                 #IMPLIED
  object        CDATA                 #IMPLIED
  alt           %Text;                #IMPLIED
  name          CDATA                 #IMPLIED
  width         %Length;              #REQUIRED
  height        %Length;              #REQUIRED
  %IAlign.attrib;
  hspace        %Pixels;              #IMPLIED
  vspace        %Pixels;              #IMPLIED
>
```

There's no indication of what the value of all these attributes should be. An additional comment like this would be helpful:

```
<!-- ATTLIST applet
  codebase  the URI where of the directory from which the
             applet is downloaded; defaults to the URI of the
             document containing the applet tag
  archive   the name of the JAR file that contains the applet;
             omitted if the applet isn't stored in a JAR
             archive
  code      the name of the main class of the applet
  object    the name of the serialized object that contains
             the main applet class; must match the name of the
             class in the applet attribute
  alt       text displayed if the applet cannot be located
  name      the name of the applet
  width     width of the applet in pixels
  height    height of the applet in pixels
  align     bottom, middle, top, left, or right
             meaning the bottom, middle, or top of the applet
             is aligned with the baseline or that the
             applet floats to the left or the right
  hspace    number of pixels with which
             to pad the left and right sides of the applet
  vspace    number of pixels with which
             to pad the top and bottom of the applet
-->
```

Of course all this could be found out by reading the specification for HTML 4.0. However, many times when complete documentation is left to a later, prose

document, that prose document never gets written. It certainly doesn't hurt to include extra commentary when you're actually writing the DTD for the first time.

Part of the problem is that restrictions on attribute values are not well expressed in DTDs; for instance that the `height` and `width` must be integers. In the future, this shortcoming may be addressed by a schema language layered on top of standard XML syntax.

In cases of complicated attribute and element declarations, it's also often useful to provide an example in a comment. For instance:

```
<!--  
    <applet width="500" height="500"  
        codebase="http://www.site.com/directory/subdirectory/"  
        archive="MyApplet.jar"  
        code="MyApplet.class"  
        object="MyApplet.ser"  
        name="FirstInstance"  
        align="top"  
        hspace="5"  
        vspace="5"  
    >  
    <param name="name1" value="value1"/>  
    <param name="name2" value="value2"/>  
    Some text for browsers that don't understand the  
    applet tag  
</applet>  
-->
```

Parameter Entities

The XHTML DTD makes extremely heavy use of both internal and external parameter entities. Your DTDs can, too. There are many uses for parameter entities that were demonstrated in the XHTML DTD. In summary, you can use them to:

- ♦ Break up long content models and attribute lists into manageable, related pieces
- ♦ Standardize common sets of elements and attributes
- ♦ Enable different DTDs to change content models and attribute lists
- ♦ Better document content models
- ♦ Compress the DTD by reusing common sequences of text
- ♦ Split the DTD into individual, related modules

Break Up Long Content Models and Attribute Lists into Manageable, Related Pieces

A typical HTML element like `p` can easily have 30 or more possible attributes and dozens of potential children. Listing them all in a content model or attribute list will simply overwhelm anyone trying to read a DTD. To the extent that related elements and attributes can be grouped, it's better to separate them into several parameter entities. For example, here's XHTML's element declaration for `p`:

```
<!ELEMENT p %P.content; >
```

It uses only a single parameter entity reference, rather than the many separate element names that the reference resolves into.

Here's XHTML's attribute list for `p`:

```
<!ATTLIST p
    %Common.attrib;
>
```

It uses only one-parameter entities rather than the many separate attribute names and content types they resolve into.

Standardize Common Sets of Elements and Attributes

When you're dealing with 30 or more items in a list, it's easy to miss one if you have to keep repeating the list. For instance, almost all HTML elements can have these attributes:

```
id class style title lang xml:lang dir onclick ondblclick
onmousedown onmouseup onmousemove onmouseout onkeypress
onkeydown onkeyup onclick ondblclick onmousedown onmouseup
onmouseover onmousemove onmouseout onkeypress onkeydown onkeyup
```

By combining them all into one `%Common.attrib;` parameter entity reference, you avoid the chance of omitting or mis-typing one of them in an attribute list. If at any point in the future, you want to add an attribute to this list, you can add it just by adding it to the declaration of `Common.attrib`. You don't have to add it to each of a hundred or more element declarations.

Enable Different DTDs to Change Content Models and Attribute Lists

One of the neatest tricks with parameter entity references in XHTML is how they're used to customize three different DTDs from the same basic modules. The key is that each customizable item, whether a content model or an attribute list, is given as a parameter entity reference. Each DTD can then redefine the content model or attribute list by redefining the parameter entity reference. This allows particular DTDs to both add and remove items from content models and attribute lists.

For example, in the XHTML1-table module, the `caption` element is defined like this:

```
<!ENTITY % Caption.content "( #PCDATA | %Inline.mix; )*" >
<!ELEMENT caption %Caption.content; >
<!ATTLIST caption
    %Common.attrib;
>
```

Suppose your DTD requires that captions only contain unmarked-up PCDATA. Then it is easy to place this entity definition in the file that imports XHTML1-table.mod:

```
<!ENTITY % Caption.content "( #PCDATA )" >
```

This will override the declaration in XHTML1-table.mod so that captions adhering to your DTD can only include text and no mark up.

Better Document Content Models

One of the most unusual tricks the XHTML DTD plays with parameter entity references is using them to replace the CDATA attribute type. Although `%ContentType;`, `%ContentTypes;`, `%Charset;`, `%Charsets;`, `%LanguageCode;`, `%Character;`, `%Number;`, `%LinkTypes;`, `%MediaDesc;`, and `%URI;`, are on one level just synonyms for CDATA, on another level they make the attribute types a lot more specific. CDATA can really mean almost anything. Using parameter entities in this way goes a long way toward narrowing down and documenting the actual meaning in a particular context. While such parameter entities can't enforce their meanings, simply documenting them is no small achievement.

Compress the DTD by Reusing Common Sequences of Text

The XHTML DTD occupies just about 80 kilobytes. That's not a huge amount, especially for applications that reside on a local drive or network, but it is non-trivial for Internet applications. It would probably be three to five times larger if all the parameter entity references were fully expanded.

Even more significant than the file size saving achieved by parameter entity references are the savings in legibility. Short files are easier to read and comprehend. A 600-kilobyte DTD, even broken up into 60-kilobyte chunks, would be too much to ask document authors to read, especially given the turgid, non-English code that makes up DTDs. (Let me put it this way: Of the much smaller modules in this chapter, how many of them did you actually read from start to finish and how many did you just skip over until the example was done? Any code module that's longer than a page is likely to thwart all but the most determined and conscientious readers.)

Split the DTD into Individual, Related Modules

On a related note, splitting the DTD into several related modules makes it easier to grasp overall. All the forms material is conveniently gathered in one place, as is all the tables material, all the applet material, and so forth. Furthermore, this makes the DTD easier to understand because you can take it one bite-sized piece at a time.

On the other hand, the interconnections between some of the modules do make this a little more confusing than perhaps it needs to be. In order to truly understand any one of the modules, you must understand the XHTML1-names.mod and XHTML1-attrs.mod because these provide crucial definitions for entities used in all the other modules. Furthermore, a module can only really be understood in the context of either the strict, loose, or frameset DTD. So there are four files you need to grasp before you can really start to get a handle on any one. Still, the clean separation between modules is impressive, and recommends itself for imitation.

Summary

In this chapter, you learned:

- ♦ All writers learn by reading other writers' work. XML writers should read other XML writers' work.
- ♦ The XHTML DTD is an XMLized version of HTML that comes in three flavors: strict, loose, and frameset.
- ♦ The XHTML DTD divides HTML into 29 different modules and three entity sets.
- ♦ You can never have too many comments in your DTDs, which make the file much easier to read.
- ♦ Parameter entities are extremely powerful tools for building complex yet manageable DTDs.

In the next chapter, we'll explore another XML application, the Channel Definition Format (CDF), used to push content to subscribers. Whereas we've concentrated almost completely on the XHTML DTD in this chapter, CDF does not actually have a published DTD, so we'll take a very different approach to understanding it.



