XHTML™ 2.0

W3C Working Group Note 16 December 2010

This version:
http://www.w3.org/TR/2010/NOTE-xhtml2-20101216
Latest public version:
http://www.w3.org/TR/xhtml2
Previous version:
http://www.w3.org/TR/2006/WD-xhtml2-20060726
Diff-marked version:
xhtml2-diff.html
Editors:
Mark Birbeck, Sidewinder Labs
Markus Gylling, DAISY Consortium
Shane McCarron, Applied Testing and Technology
Steven Pemberton, CWI (XHTML 2 Working Group Co-Chair)
Editors (while chartered as the HTML Working Group):
Jonny Axelsson, Opera Software
Mark Birbeck, Sidewinder Labs
Micah Dubinko, Invited Expert
Beth Epperson, Websense
Masayasu Ishikawa, W3C
Shane McCarron, Applied Testing and Technology
Ann Navarro, Invited Expert
Steven Pemberton, CWI

This document is also available in these non-normative formats: Single XHTML file [p.1] , PostScript version, PDF version, ZIP archive, and Gzip'd TAR archive.

Copyright © 2001-2010 W3C® (MIT, ERCIM, Keio), All Rights Reserved. W3C liability, trademark and document use rules apply.

Abstract

XHTML 2 is a general-purpose markup language designed to represent documents for a wide range of purposes across the World Wide Web. To this end it does not attempt to be all things to all people, supplying every possible markup idiom, but to supply a generally useful set of elements.

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This document is a Working Group Note. The XHTML2 Working Group's charter expired before it could complete work on this document. It is possible that the work will be continued by another group in the future.

Publication as a Working Group Note does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

This document has been produced by the W3C XHTML 2 Working Group as part of the HTML Activity. The goals of the XHTML 2 Working Group are discussed in the XHTML 2 Working Group charter.

This document was produced by a group operating under the 5 February 2004 W3C Patent Policy. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Public discussion of HTML takes place on www-html@w3.org (archive). To subscribe send an email to www-html-request@w3.org with the word *subscribe* in the subject line.

Please report errors in this document to www-html-editor@w3.org (archive).

Quick Table of Contents

1. Introduction
2. Terms and Definitions 17
3. Conformance Definition
4. The XHTML 2.0 Document Type 25
5. Module Definition Conventions 27
6. XHTML Attribute Collections 33
7. XHTML Document Module 35
8. XHTML Structural Module 39
9. XHTML Text Module 47
10. XHTML Hypertext Module 55
11. XHTML List Module 57
12. XHTML Core Attributes Module 61
13. XHTML Hypertext Attributes Module 65

14. XHTML I18N Attribute Module 71
15. XHTML Access Module 73
16. XHTML Bi-directional Text Attribute Module 79
17. XHTML Caption Module 81
18. XHTML Edit Attributes Module 83
19. XHTML Embedding Attributes Module 85
20. XHTML Image Module 87
21. XHTML Image Map Attributes Module 89
22. XHTML Media Attribute Module 93
23. XHTML Metainformation Module 95
24. XHTML Metainformation Attributes Module 99
25. XHTML Object Module 101
26. XHTML Role Attribute Module 111
27. Ruby Module 113
28. XHTML Style Sheet Module 115
29. XHTML Style Attribute Module 119
30. XHTML Tables Module 121
31. XForms Module 143
32. XML Events Module 147
33. XML Handlers Module 149
34. XML Scripting Module 151
35. XHTML Legacy Edit Module 153
36. XHTML Legacy Headings Module 155
A. Changes from earlier XHTML versions 157
B. XHTML 2.0 RELAX NG Definition 159
C. XHTML RELAX NG Module Implementations 163
D. XHTML 2.0 Schema 241
E. XHTML Schema Module Implementations 253
F. XHTML 2.0 Document Type Definition 291
G. XHTML DTD Module Implementations 293
H. Style sheet for XHTML 2 295
I. List of Elements 299
J. List of Attributes 303
K. Cross-reference Index 307
L. References 309
M. Acknowledgements 313

Full Table of Contents

1. Introduction
    1.1. What is XHTML 2?
        1.1.1. Design Aims
        1.1.2. Backwards compatibility

1.1.3. XHTML 2 and Presentation
1.1.4. XHTML 2 and Linking
1.2. Major Differences with XHTML 1
1.3. What are the XHTML 2 Modules?
2. Terms and Definitions 17
3. Conformance Definition
3.1. Document Conformance
3.1.1. Conforming Documents
3.2. XHTML 2 User Agent Conformance
4. The XHTML 2.0 Document Type 25
5. Module Definition Conventions 27
5.1. Module Structure 27
5.2. Abstract Module Definitions 27
5.3. Syntactic Conventions 27
5.4. Content Models 28
5.5. Attribute Types 29
6. XHTML Attribute Collections 33
7. XHTML Document Module 35
7.1. The html element 35
7.2. The head element 36
7.3. The title element 37
7.4. The body element 37
8. XHTML Structural Module 39
8.1. The address element 40
8.2. The blockcode element 40
8.3. The blockquote element 41
8.4. The div element 41
8.5. The h element 42
8.6. The p element 43
8.7. The pre element 43
8.8. The section element 44
8.9. The separator element 45
9. XHTML Text Module 47
9.1. The abbr element 48
9.2. The br element 48
9.3. The cite element 49
9.4. The code element 49
9.5. The dfn element 50
9.6. The em element 50
9.7. The kbd element 50
9.8. The l element 51
9.9. The q element 51
9.10. The samp element 52

9.11. The span element 52
9.12. The strong element 53
9.13. The sub element 53
9.14. The sup element 53
9.15. The var element 54
10. XHTML Hypertext Module 55
10.1. The a element 55
11. XHTML List Module 57
11.1. Definition lists: the dl, di, dt, and dd elements 58
11.2. The ol and ul elements 59
11.3. The li element 60
12. XHTML Core Attributes Module 61
12.1. Core Attribute Collection 61
13. XHTML Hypertext Attributes Module 65
13.1. Hypertext Attribute Collection 65
14. XHTML I18N Attribute Module 7.1
14.1. I18N Attribute Collection 7.1
15. XHTML Access Module 7.3
15.1. The access element 7.3
15.1.1. Examples 7.6
16. XHTML Bi-directional Text Attribute Module 7.9
16.1. Bi-directional Text Collection 7.9
16.1.1. Inheritance of text direction information 7.9
16.1.2. The effect of style sheets on bidirectionality 80
17. XHTML Caption Module 81
17.1. The caption element 81
18. XHTML Edit Attributes Module 83
18.1. Edit Collection 83
19. XHTML Embedding Attributes Module 85
19.1. Embedding Attribute Collection 85
20. XHTML Image Module 87
20.1. The img element 87
21. XHTML Image Map Attributes Module 89
21.1. Image Map Attribute Collection 89
22. XHTML Media Attribute Module 93
22.1. Media Attribute Collection 93
23. XHTML Metainformation Module 95
23.1. The link element 95
23.1.1. Forward and reverse links 96
23.1.2. Links and search engines 96
23.2. The meta element 97
23.2.1. meta and search engines 98
24. XHTML Metainformation Attributes Module 99

24.1. Metadata Attribute Collection 99
25. XHTML Object Module 101
25.1. The object element 101
25.1.1. Defining terminology 102
25.1.2. Basic Information for Object Handlers 102
25.1.3. Rules for processing objects 103
25.2. The param element 106
25.2.1. Referencing object data 108
25.2.2. Object element declarations and instantiations 108
25.3. The standby element 110
26. XHTML Role Attribute Module 111
26.1. Role Attribute Collection 111
27. Ruby Module 113
28. XHTML Style Sheet Module 115
28.1. The style element 115
28.1.1. External style sheets 116
28.1.2. Preferred and alternate style sheets 116
28.1.3. Specifying external style sheets 116
29. XHTML Style Attribute Module 119
29.1. Style Attribute Collection 119
30. XHTML Tables Module 121
30.1. The col and colgroup elements 122
30.1.1. Calculating the number of columns in a table 124
30.2. The summary element 124
30.3. The table element 125
30.3.1. Visual Rendering 125
30.3.2. Table directionality 126
30.3.3. Table rendering by non-visual user agents 126
30.4. The tbody element 135
30.5. The td and th elements 135
30.5.1. Cells that span several rows or columns 137
30.6. The thead and tfoot elements 140
30.7. The tr element 141
31. XForms Module 143
31.1. Core XForms 143
31.2. XForms Actions 143
31.3. Form Controls 143
31.4. Group 144
31.5. Switch 144
31.6. Repeat 144
31.7. XForms Repeat Attribute Collection 144
31.8. Other Attribute Collections 145
32. XML Events Module 147

32.1. Events 147
33. XML Handlers Module 149
34. XML Scripting Module 151
35. XHTML Legacy Edit Module 153
35.1. The del element 153
35.2. The ins element 154
36. XHTML Legacy Headings Module 155
36.1. The heading elements 155
A. Changes from earlier XHTML versions 157
B. XHTML 2.0 RELAX NG Definition 159

B.0.1. RELAX NG XHTML 2.0 Driver 159
C. XHTML RELAX NG Module Implementations 163
C.1. XHTML Module Implementations 163
C.1.1. Attribute Collections 163
C.1.2. Document 164
C.1.3. Structural 166
C.1.4. Text 170
C.1.5. Hypertext 175
C.1.6. List 175
C.1.7. Core Attributes 178
C.1.8. Hypertext Attributes 180
C.1.9. I18N Attributes 181
C.1.10. Access 182
C.1.11. Bi-directional Text Attribute 183
C.1.12. Edit Attributes 183
C.1.13. Embedding Attributes 184
C.1.14. Image 185
C.1.15. Image Map Attributes 185
C.1.16. Media Attribute 186
C.1.17. Metainformation Attributes 186
C.1.18. Metainformation 188
C.1.19. Object 189
C.1.20. Style Attribute 190
C.1.21. Style Sheet 190
C.1.22. Tables 191
C.2. XHTML RELAX NG Support Modules 195
C.2.1. Datatypes 196
C.2.2. Events 199
C.2.3. Param 201
C.2.4. Caption 202
C.2.5. Role Attribute 203
C.3. XHTML RELAX NG Legacy Modules 203

C.3.1. Legacy Edit 203
C.3.2. Legacy Headings 204
C.3.3. Legacy Line Break 205
C.4. RELAX NG External Modules 205
C.4.1. Ruby 205
C.4.2. Ruby Driver for Full Ruby Markup 208
C.4.3. XML Events 208
C.4.4. XML Handlers 210
C.4.5. XML Script 212
C.4.6. XML Schema instance 214
C.4.7. XForms 1.1 215
C.4.8. XForms Repeat Attribute Collection 238
D. XHTML 2.0 Schema 241
D.1. XHTML 2 Schema Driver 241
D.2. XHTML 2 Attribute Collections 243
D.3. XHTML 2 Content Model 244
D.4. XHTML 2 Modules 250
E. XHTML Schema Module Implementations 253
E.1. Required Modules 253
E.1.1. Datatypes Module 253
E.1.2. Document Module 255
E.1.3. Structural Module 257
E.1.4. Text Module 260
E.1.5. Hypertext Module 264
E.1.6. List Module 264
E.1.7. Core Attributes Module 266
E.1.8. Hypertext Attributes Module 267
E.1.9. I18N Attribute Module 268
E.2. Optional Modules 269
E.2.1. Bi-directional Text Module 269
E.2.2. Caption Module 269
E.2.3. Edit Attributes Module 270
E.2.4. Embedding Attributes Module 270
E.2.5. Image Module 271
E.2.6. Image Map Attributes Module 272
E.2.7. Media Attribute Module 272
E.2.8. Metainformation Module 273
E.2.9. Metainformation Attributes Module 274
E.2.10. Object Module 274
E.2.11. Style Sheet Module 276
E.2.12. Style Attribute Module 277
E.2.13. Tables Module 277
E.3. Legacy Module 279

E.3.1. Legacy Line Break Module 279
E.3.2. Legacy Editing Module 280
E.3.3. Legacy Headings Module 280
E.4. Modules from Other Specifications 282
E.4.1. Access Module 282
E.4.2. Role Attribute Module 283
E.4.3. Ruby Module 283
E.4.4. XForms Modules 286
E.4.5. XML Events Module 286
E.4.6. XML Handlers Module 287
E.4.7. XML Scripting Module 289
F. XHTML 2.0 Document Type Definition 291
G. XHTML DTD Module Implementations 293
G.1. XHTML Modular Framework 293
G.2. XHTML Module Implementations 293
G.3. XHTML DTD Support Modules 293
H. Style sheet for XHTML 2 295
I. List of Elements 299
J. List of Attributes 303
K. Cross-reference Index 307
L. References 309
L.1. Normative References 309
L.2. Informative References 312
M. Acknowledgements 313

1. Introduction

This section is *informative*.

1.1. What is XHTML 2?

XHTML 2 is a general purpose markup language designed to represent documents for a wide range of purposes across the World Wide Web. To this end it does not attempt to be all things to all people, supplying every possible markup idiom, but to supply a generally useful set of elements, with the possibility of extension using the @class [p.61] and @role [p.111] attributes on the span [p.52] and div [p.41] elements in combination with style sheets, and attributes from the metadata attributes collection.

1.1.1. Design Aims

In designing XHTML 2, a number of design aims were kept in mind to help direct the design. These included:

- As generic XML as possible: if a facility exists in XML, try to use that rather than duplicating it.
- Less presentation, more structure: use style sheets for defining presentation.
- More usability: within the constraints of XML, try to make the language easy to write, and make the resulting documents easy to use.
- More accessibility: some call it 'designing for our future selves' – the design should be as inclusive as possible.
- Better internationalization: since it is a World Wide Web.
- More device independence: new devices coming online, such as telephones, PDAs, tablets, televisions and so on mean that it is imperative to have a design that allows you to author once and render in different ways on different devices, rather than authoring new versions of the document for each type of device.
- Less scripting: achieving functionality through scripting is difficult for the author and restricts the type of user agent you can use to view the document. We have tried to identify current typical usage, and include those usages in markup.
- Integration with the Semantic Web: make XHTML2 amenable for processing with semantic web tools.

1.1.2. Backwards compatibility

Because earlier versions of HTML were special-purpose languages, it was necessary to ensure a level of backwards compatibility with new versions so that new documents would still be usable in older browsers. However, thanks to XML and style sheets, such strict element-wise backwards compatibility is no longer necessary, since an XML-based browser, of which at the time of writing means more than 95% of browsers in use, can process new markup languages without having to be updated. Much of XHTML 2 works already in existing browsers; much, but not all: just as when forms and tables were added to HTML, and people had to wait for new

version of browsers before being able to use the new facilities, some parts of XHTML 2, principally XForms and XML Events, still require user agents that understand that functionality.

1.1.3. XHTML 2 and Presentation

The very first version of HTML was designed to represent the structure of a document, not its presentation. Even though presentation-oriented elements were later added to the language by browser manufacturers, HTML is at heart a document structuring language. XHTML 2 takes HTML back to these roots, by removing all presentation elements, and subordinating all presentation to style sheets. This gives greater flexibility, greater accessibility, more device independence, and more powerful presentation possibilities, since style sheets can do more than the presentational elements of HTML ever did.

1.1.4. XHTML 2 and Linking

The original versions of HTML relied upon built-in knowledge on the part of User Agents and other document processors. While much of this knowledge had to do with presentation (see above), the bulk of the remainder had to do with the relationships between documents — so called "linking".

A variety of W3C and other efforts, most notably [XLINK [p.312]], attempted to create a grammar for defining the characteristings of linking. Unfortunately, these grammars all fall short of the requirements of XHTML. The community is continuing in its efforts to create a comprehensive grammar that describes link characteristics.

The HTML Working Group has determined that such a grammar, while generally useful, is not required for the definition of XHTML 2. Instead, this document is explicit in the characteristics of the elements and attributes that are used to connect to other resources. The Working Group has taken this course because 1) the problem with XHTML 2 is well bounded, 2) the general solution is slow in coming, and 3) it will be easier for implementors to support and users to rely upon.

1.2. Major Differences with XHTML 1

XHTML 2 is designed to be recognizable to the HTML and XHTML 1 author, while correcting errors and insufficiencies identified in earlier versions of the HTML family, and taking the opportunity to make improvements.

The most visible changes are the following:

- More structuring possibilities:
  - Sections and headings: in previous versions of HTML a document's structure had to be inferred from the various levels of headings in the document; this was particularly a problem when authors misused the heading elements for visual effects. XHTML 2 lets you explicitly markup the document structure with the section [p.44] element, and its related header element h [p.42] .
  - Separators: in previous versions of HTML, the `hr` element was used to separate

sections of a text from each other. In retrospect, the name `hr` (for *horizontal rule*) was misleading, because an `hr` was neither necessarily horizontal (in vertical text it was vertical), nor necessarily a rule (books often use other typographical methods such as a line of three asterisks to represent separators, and style sheets can be used to give you this freedom). In order to emphasize its structuring nature, to make it more widely usable, and to make it clearer that it has no essential directionality, `hr` has been renamed separator [p.45] .

- Line breaks: in previous versions of HTML, the `br` element was used to add micro-structure to text, essentially breaking a piece of text into several 'lines'. This micro-structure is now made explicit in XHTML 2 with the l [p.51] element, which encloses the text to be broken. Amongst other advantages, this gives more presentational opportunities, such as the ability to automatically number lines, or to color alternate lines differently.
- Paragraph structure: in earlier versions of HTML, a p [p.43] element could only contain simple text. It has been improved to bring it closer to what people perceive as a paragraph, now being allowed to include such things as lists and tables.

- Images: the HTML `img` element has many shortcomings: it only allows you to specify a single resource for an image, rather than offering the fallback opportunities of the object [p.101] element; the only fallback option it gives is the `alt` text, which can only be plain text, and not marked up in any way; the `longdesc` attribute which allows you to provide a long description of the image is difficult to author and seldom supported.

  XHTML 2 takes a completely different approach, by taking the premise that all images have a long description and treating the image and the text as equivalents. In XHTML 2 *any* element may have a @src [p.86] attribute, which specifies a resource (such as an image) to load instead of the element. If the resource is unavailable (because of network failure, because it is of a type that the browser can't handle, or because images have been turned off) then the element is used instead. Essentially the `longdesc` has been moved into the document, though this behavior also mimicks the fallback behavior of the object [p.101] element. (To achieve the tooltip effect that some browsers gave with the `alt` attribute, as in HTML 4 you use the @title [p.62] attribute).

- Type: in HTML 4, the @srctype [p.86] attribute when referring to an external resource was purely a hint to the user agent. In XHTML 2 it is no longer a hint, but specifies the type(s) of resource the user agent must accept.
- Tables: the content model of tables has been cleaned up and simplified, while still allowing the same functionality.
- Bi-directional text: rather than use an explicit element to describe bi-directional override, new values have been added to the @dir [p.79] attribute that allow bi-directional override on any element.
- Edit: rather than use explicit `ins` and `del` elements to mark changes in a document, an attribute @edit [p.83] may be used on any element for the same purpose.
- Linking: In HTML 3.2, only a [p.55] elements could be the source and target of hyperlinks. In HTML 4 and XHTML 1, any element could be the target of a hyperlink, but still only a [p.55] elements could be the source. In XHTML 2 any element can now also be the source of a hyperlink, since @href [p.65] and its associated attributes may now appear on any element.

So for instance, instead of `<li><a href="home.html">Home</a></li>`, you can now write `<li href="home.html">Home</li>`. Even though this means that the a [p.55] element is now strictly-speaking unnecessary, it has been retained.

- Metadata: the meta [p.97] and link [p.95] elements have been generalized, and their relationship to RDF [RDF [p.312]] described. Furthermore, the attributes on these two elements can be more generally applied across the language.
- Role: in order to aid adding semantics to documents, the @role [p.111] attribute has been added, along with an initial set of useful values, in order to classify the use of a particular element. For instance a paragraph may play the role of a note, and so may be marked up `<p role="note">`.

- Events: event handling in HTML was restricted in several ways: since the event names were hard-wired in the language (such as `onclick`), the only way to add new events was to change the language; many of the events (such as `click`) were device-specific, rather than referring to the intent of the event (such as activating a link); you could only handle an event in one scripting language — it was not possible to supply event handlers in the same document for several different scripting languages.

  XHTML 2 uses XML Events [XMLEVENTS [p.311]] to specify event handling, giving greater freedom in the ability to handle events. Along with this, the `script` element has been renamed `handler` to indicate its different semantics.

- Forms: HTML Forms were introduced in 1993, before the advent of the e-commerce revolution. Now with more than a decade of experience with their use, they have been thoroughly overhauled and updated to meet the needs of modern forms, in the shape of XForms [XFORMS [p.311]], which are an integral part of XHTML 2.
- Ownership where due: since HTML 4 was a standalone application, it defined many things which no longer need to be defined now that it is an XML application. For instance the definitions of whitespace are given by XML for input, and CSS for output; similarly, the definition of values of the @media [p.115] attribute are relegated to the relevant style sheet language.
- Frames and Framesets: In HTML 4 multi-panel "pages" could be described using the `frameset` and `frame` elements. The Frames model is no longer defined in XHTML. Instead, it is defined through the separate [XFRAMES [p.312]] specification.

1.3. What are the XHTML 2 Modules?

XHTML 2 is a member of the XHTML Family of markup languages. It is an XHTML Host Language as defined in XHTML Modularization. As such, it is made up of a set of XHTML Modules that together describe the elements and attributes of the language, and their content model. XHTML 2 updates many of the modules defined in XHTML Modularization 1.1 [XHTMLMOD [p.311]], and includes the updated versions of all those modules and their semantics. XHTML 2 also uses modules from XHTML Access [XHTMLACCESS [p.311]], XHTML Role [XHTMLROLE [p.311]], RDFa Syntax [RDFASYNTAX [p.310]], Ruby [RUBY [p.310]], XML Events [XMLEVENTS [p.311]], and XForms [XFORMS [p.311]].

The modules defined in this specification are largely extensions of the modules defined in XHTML Modularization 1.1. This specification also defines the semantics of the modules it includes. So, that means that unlike earlier versions of XHTML that relied upon the semantics defined in HTML 4 [HTML4 [p.312]], all of the semantics for XHTML 2 are defined either in this specification or in the specifications that it normatively references.

Even though the XHTML 2 modules are defined in this specification, they are available for use in other XHTML family markup languages. Over time, it is possible that the modules defined in this specification will migrate into an update of the XHTML Modularization specification.

2. Terms and Definitions

This section is *normative*.

While some terms are defined in place, the following definitions are used throughout this document. Familiarity with the W3C XML 1.0 Recommendation [XML [p.311]] is highly recommended.

abstract module
: a unit of document type specification corresponding to a distinct type of content, corresponding to a markup construct reflecting this distinct type.

content model
: the declared markup structure allowed within instances of an element type. XML 1.0 differentiates two types: elements containing only element content (no character data) and mixed content (elements that may contain character data optionally interspersed with child elements). The latter are characterized by a content specification beginning with the "#PCDATA" string (denoting character data).

deprecated
: a feature marked as deprecated is in the process of being removed from this recommendation. Portable documents should not use features marked as deprecated.

document model
: the effective structure and constraints of a given document type. The document model constitutes the abstract representation of the physical or semantic structures of a class of documents.

document type
: a class of documents sharing a common abstract structure. The ISO 8879 [SGML [p.310]] definition is as follows: "a class of documents having similar characteristics; for example, journal, article, technical manual, or memo. (4.102)"

document type definition (DTD)
: a formal, machine-readable expression of the XML structure and syntax rules to which a document instance of a specific document type must conform; the schema type used in XML 1.0 to validate conformance of a document instance to its declared document type. The same markup model may be expressed by a variety of DTDs.

driver
: a generally short file used to declare and instantiate the modules of a DTD. A good rule of thumb is that a DTD driver contains no markup declarations that comprise any part of the document model itself.

element
: an instance of an element type.

element type
: the definition of an element, that is, a container for a distinct semantic class of document content.

entity
: an entity is a logical or physical storage unit containing document content. Entities may be composed of parseable XML markup or character data, or unparsed (i.e., non-XML, possibly non-textual) content. Entity content may be either defined entirely within the

document entity ("internal entities") or external to the document entity ("external entities"). In parsed entities, the replacement text may include references to other entities.

entity reference
: a mnemonic string used as a reference to the content of a declared entity (e.g., "&" for "&", "<" for "<", "©" for "©".)

facilities
: Facilities are elements, attributes, and the semantics associated with those elements and attributes.

focusable
: Elements are considered "focusable" if they are *visible* (e.g., have the equivalent of the [CSS2 [p.309]] property of "display" with a value other than `none`) not disabled (see [XFORMS [p.311]]), and either 1) have an @href [p.65] attribute or 2) are considered a form control as defined in [XFORMS [p.311]].

fragment identifier
: A portion of a [URI [p.311]] as defined in RFC 3986.

generic identifier
: the name identifying the element type of an element. Also, element type name.

hybrid document
: A hybrid document is a document that uses more than one XML namespace. Hybrid documents may be defined as documents that contain elements or attributes from hybrid document types.

instantiate
: to replace an entity reference with an instance of its declared content.

markup declaration
: a syntactical construct within a DTD declaring an entity or defining a markup structure. Within XML DTDs, there are four specific types: entity declaration defines the binding between a mnemonic symbol and its replacement content; element declaration constrains which element types may occur as descendants within an element (see also content model); attribute definition list declaration defines the set of attributes for a given element type, and may also establish type constraints and default values; notation declaration defines the binding between a notation name and an external identifier referencing the format of an unparsed entity.

markup model
: the markup vocabulary (i.e., the gamut of element and attribute names, notations, etc.) and grammar (i.e., the prescribed use of that vocabulary) as defined by a document type definition (i.e., a schema) The markup model is the concrete representation in markup syntax of the document model, and may be defined with varying levels of strict conformity. The same document model may be expressed by a variety of markup models.

module
: an abstract unit within a document model expressed as a DTD fragment, used to consolidate markup declarations to increase the flexibility, modifiability, reuse and understanding of specific logical or semantic structures.

modularization
: an implementation of a modularization model; the process of composing or de-composing a DTD by dividing its markup declarations into units or groups to support specific goals. Modules may or may not exist as separate file entities (i.e., the physical and logical

structures of a DTD may mirror each other, but there is no such requirement).

modularization model
: the abstract design of the document type definition (DTD) in support of the modularization goals, such as reuse, extensibility, expressiveness, ease of documentation, code size, consistency and intuitiveness of use. It is important to note that a modularization model is only orthogonally related to the document model it describes, so that two very different modularization models may describe the same document type.

parameter entity
: an entity whose scope of use is within the document prolog (i.e., the external subset/DTD or internal subset). Parameter entities are disallowed within the document instance.

parent document type
: A parent document type of a hybrid document is the document type of the root element.

tag
: descriptive markup delimiting the start and end (including its generic identifier and any attributes) of an element.

unavailable resource
: any resource that is referenced as a URI in an attribute, but that cannot be accessed for any reason, is considered unavailable. Example reasons include, but are not limited to: network unavailable, no resource available at the URI given, inability of the user agent to process the type of resource, etc.

user agent
: any software that retrieves and renders Strictly Conforming Documents for users. This may include browsers, media players, plug-ins, and other programs — including assistive technologies — that help in retrieving and rendering such documents. See also Conforming User Agent.

3. Conformance Definition

This section is *normative*.

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119 [p.310]].

3.1. Document Conformance

In this document, the use of the word 'schema' refers to any definition of the syntax of XHTML 2, regardless of the definition language used.

3.1.1. Conforming Documents

A conforming XHTML 2.0 document is a document that requires only the facilities described as mandatory in this specification. Such a document must meet all the following criteria:

1. The document must conform to the constraints expressed in the prose throughout this document, and also to the machine-checkable constraints expressed in the schema in Appendix B - XHTML 2.0 RELAX NG Definition [p.159] .

2. The local part of the root element of the document must be `html`.

3. The start tag of the root element of the document must explicitly contain an `xmlns` declaration for the XHTML namespace [XMLNS [p.311]]. The namespace URI for XHTML is defined to be `http://www.w3.org/1999/xhtml`.

   The start tag may also contain an @xsi:schemaLocation [p.36] attribute that associates this namespace with the XML Schema at the URI `http://www.w3.org/MarkUp/SCHEMA/xhtml2.xsd`.

   Sample root element

   ```
   <html xmlns="<xmlns>" xml:lang="en"
         xmlns:xsi="<xsilocation>"
         xsi:schemaLocation="<xmlns>
                             <xmlschemaloc>"
   >
   ```

4. There MAY be a DOCTYPE declaration in the document prior to the root element. If present, the public identifier included in the DOCTYPE declaration MUST reference the DTD found in Appendix F [p.293] using its Public Identifier. The system identifier may be modified appropriately.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 2.0//EN"
    "http://www.w3.org/MarkUp/DTD/xhtml2.dtd">
```

Example of an XHTML 2.0 document

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="<xmlns />" xml:lang="en">
  <head>
    <title>Virtual Library</title>
  </head>
  <body>
    <p>Moved to <a href="http://example.org/">example.org</a>.</p>
  </body>
</html>
```

Note that in this example, the XML declaration is included. An XML declaration like the one above is not required in all XML documents. XHTML document authors should use XML declarations in all their documents. XHTML document authors must use an XML declaration when the character encoding of the document is other than the default UTF-8 or UTF-16 and no encoding is specified by a higher-level protocol.

3.2. XHTML 2 User Agent Conformance

A conforming user agent must meet all of the following criteria, as well as other requirements found throughout this specification:

1. The user agent must parse and evaluate an XHTML 2 document for well-formedness. If the user agent claims to be a validating user agent, it must also validate documents against a referenced schema according to [XML [p.311]].

2. When the user agent claims to support facilities defined within this specification or required by this specification through normative reference, it must do so in ways consistent with the facilities' definition.

3. A user agent must only recognize attributes of type `ID` (e.g., the `id` or `xml:id` attribute on most XHTML 2 elements) as fragment identifiers.

4. If a user agent encounters an element it does not recognize, it must continue to process the content of that element. If this "unrecognized" element uses recognized attributes, those attributes must be processed with their usual semantics.

5. If a user agent encounters an attribute it does not recognize, it must ignore the entire attribute specification (i.e., the attribute and its value).

6. If a user agent encounters an attribute value it doesn't recognize, it must use the default attribute value.

7. When rendering content, user agents that encounter characters or character entity references that are recognized but not renderable should display the document in such a way that it is obvious to the user that normal rendering has not taken place.

8. White space must be handled according to the rules of [XML [p.311]]. All XHTML 2 elements preserve whitespace.

   The user agent must use the definition from CSS for processing white space characters [CSS3-TEXT [p.309]].

9. In the absence of a style-sheet, including user agents that do not process style sheets, the default visual presentation should be as if the user agent used the CSS style sheet specified in Appendix H.

Note that this specification does not generally specify the behavior of conforming implementations when presented with non-conforming documents. This is either defined by an underlying specification (e.g., [XML [p.311]]) or left to the implementor.

4. The XHTML 2.0 Document Type

This section is *normative*.

The XHTML 2.0 document type is a fully functional document type with rich semantics. It is a collection of XHTML-conforming modules (many of which are defined in this specification). The Modules and the elements and/or attributes they define are listed here for information purposes, but the definitions in their base documents should be considered authoritative. In the on-line version of this document, the module names in the list below link into the definitions of the modules within the relevant version of the authoritative specification.

Document Module [p.35]
    `body, head, html, title`
Structural Module [p.39]
    `address, blockcode, blockquote, div, h, h1, h2, h3, h4, h5, h6, p, pre, section, separator`
Text Module [p.47]
    `abbr, cite, code, dfn, em, kbd, l, q, samp, span, strong, sub, sup, var`
Hypertext Module [p.55]
    `a`
List Module [p.57]
    `dl, dt, dd, label, nl, ol, ul, li`
Core Attributes Module [p.61]
    `class`, `id`, and `title` attributes
Hypertext Attributes Module [p.65]
    `href`, `hreftype`, `cite`, `target`, `rel`, `rev`, `access`, `nextfocus`, `prevfocus`, and `xml:base` attributes
Internationalization Attribute Module [p.71]
    `xml:lang` attribute
Bi-directional Text Module [p.79]
    `dir` attribute
Edit Attributes Module [p.83]
    `edit` and `datetime` attributes
Embedding Attributes Module [p.85]
    `src` and `type` attributes
Image Map Attributes Module [p.89]
    `usemap, ismap, shape`, and `coords` attributes
Media Attribute Module [p.93]
    `media` attribute
Metainformation Module [p.95]
    `meta`, `link`
Object Module [p.101]
    `object, param, standby`

Style Attribute Module [p.119]
: `style` attribute

Stylesheet Module [p.115]
: `style` element

Tables Module [p.121]
: `caption`, `col`, `colgroup`, `summary`, `table`, `tbody`, `td`, `tfoot`, `th`, `thead`, `tr`

XHTML 2.0 also uses the following externally defined modules:

Ruby Annotation Module [RUBY [p.310]]
: `ruby`, `rbc`, `rtc`, `rb`, `rt`, `rp`

XForms 1.1 [XFORMS [p.311]]
: `alert`, `bind`, `case`, `choices`, `copy`, `delete`, `extension`, `filename`, `group`, `help`, `hint`, `input`, `insert`, `instance`, `item`, `itemset`, `label`, `load`, `mediatype`, `message`, `model`, `output`, `range`, `rebuild`, `recalculate`, `refresh`, `repeat`, `reset`, `revalidate`, `secret`, `select`, `select1`, `send`, `setfocus`, `setindex`, `setvalue`, `submission`, `submit`, `switch`, `textarea`, `toggle`, `trigger`, `upload`, and `value` elements, and `repeat-model`, `repeat-bind`, `repeat-nodeset`, `repeat-startindex`, and `repeat-number` attributes

XHTML Metainformation Attributes Module [RDFASYNTAX [p.310]]
: `about`, `content`, `datatype`, `typeof`, `property`, `rel`, `resource`, and `rev` attributes

XML Events Module [XMLEVENTS [p.311]]
: `listener` element, and `defaultAction`, `event`, `handler`, `objserver`, `phase`, `propagate`, and `target` attributes

XML Handlers Module [XMLEVENTS [p.311]]
: `action`, `dispatchEvent`, `addEventListener`, `removeEventListener`, `stopPropagation`, `preventDefault` attributes

XML Script Module [XMLEVENTS [p.311]]
: `script` element

Internationalization Tag Set [ITS [p.309]]
: `rules` element, `translate` attribute, etc. @@@@need to determine how to best integrate this into the content model@@@@

An implementation of this document type as a RELAX NG grammar is defined in Appendix B [p.159] , as an XML Schema in Appendix D [p.241] , and as a DTD in Appendix F [p.291] .

5. Module Definition Conventions

This section is *normative*.

This document defines a variety of XHTML modules and the semantics of those modules. This section describes the conventions used in those module definitions.

5.1. Module Structure

Each module in this document is structured in the following way:

- An abstract definition [p.27] of the module's elements, attributes, and content models, as appropriate.
- A sub-section for each element in the module; These sub-sections contain the following components:
  - A brief description of the element,
  - A definition of each attribute or attribute collection [p.33] usable with the element, and
  - A detailed description of the behavior of the element, if appropriate.

  Note that attributes are fully defined only the first time they are used in each module. After that, only a brief description of the attribute is provided, along with a link back to the primary definition.

5.2. Abstract Module Definitions

An abstract module is a definition of an XHTML module using prose text and some informal markup conventions. While such a definition is not generally useful in the machine processing of document types, it is critical in helping people understand what is contained in a module. This section defines the way in which XHTML abstract modules are defined. An XHTML-conforming module is *not required* to provide an abstract module definition. However, anyone developing an XHTML module is encouraged to provide an abstraction to ease in the use of that module.

5.3. Syntactic Conventions

The abstract modules are not defined in a formal grammar. However, the definitions do adhere to the following syntactic conventions. These conventions are similar to those of XML DTDs, and should be familiar to XML DTD authors. Each discrete syntactic element can be combined with others to make more complex expressions that conform to the algebra defined here.

element name
: When an element is included in a content model, its explicit name will be listed.

content set
: Some modules define lists of explicit element names called *content sets*. When a content set is included in a content model, its name will be listed. Content sets names always begin with an uppercase letter.

`expr ?`
: Zero or one instances of expr are permitted.

`expr +`
: One or more instances of expr are required.

`expr *`
: Zero or more instances of expr are permitted.

`a , b`
: Expression `a` is required, followed by expression `b`.

`a | b`
: Either expression a or expression b is required.

`a - b`
: Expression a is permitted, omitting elements in expression b.

parentheses
: When an expression is contained within parentheses, evaluation of any subexpressions within the parentheses take place before evaluation of expressions outside of the parentheses (starting at the deepest level of nesting first).

extending pre-defined elements
: In some instances, a module adds attributes to an element. In these instances, the element name is followed by an ampersand (`&`).

defining required attributes
: When an element requires the definition of an attribute, that attribute name is followed by an asterisk (`*`).

defining the type of attribute values
: When a module defines the type of an attribute value, it does so by listing the type in parentheses after the attribute name.

defining the legal values of attributes
: When a module defines the legal values for an attribute, it does so by listing the explicit legal values (enclosed in quotation marks), separated by vertical bars (`|`), inside of parentheses following the attribute name. If the attribute has a default value, that value is followed by an asterisk (`*`). If the attribute has a fixed value, the attribute name is followed by an equals sign (`=`) and the fixed value enclosed in quotation marks.

5.4. Content Models

Abstract module definitions define minimal, atomic content models for each module. These minimal content models reference the elements in the module itself. They may also reference elements in other modules upon which the abstract module depends. Finally, the content model in many cases requires that text be permitted as content to one or more elements. In these cases, the symbol used for text is PCDATA (parsed characted data). This is a term, defined in the XML 1.0 Recommendation, that refers to processed character data. A content type can also be defined as EMPTY, meaning the element has no content in its minimal content model.

5.5. Attribute Types

In some instances, it is necessary to define the types of attribute values or the explicit set of permitted values for attributes. The following attribute types (defined in the XML 1.0 Recommendation) are used in the definitions of the abstract modules:

Attribute Type	Definition
CDATA	Character data
ID	A document-unique identifier
IDREF	A reference to a document-unique identifier
IDREFS	A space-separated list of references to document-unique identifiers
NMTOKEN	A name composed of only name tokens as defined in XML 1.0 [XML [p.311]].
NMTOKENS	One or more white space separated NMTOKEN values
NUMBER	Sequence of one or more digits ([0-9])

In addition to these pre-defined data types, XHTML Modularization defines the following data types and their semantics (as appropriate):

Data type	Description
Character	A single character, as per section 2.2 of [XML [p.311]].
Characters	A white space separated list of Character [p.29] values.
Charset	A character encoding, as per [RFC2045] [p.310] .
Encodings	A comma-separated list of ’charset’s with optional q parameters, as defined in section 14.2 of [RFC2616 [p.310]] as the field value of the Accept-Charset request header.
ContentType	A media type, as per [RFC2045 [p.310]].

ContentTypes	Attributes of this type identify the allowable content type(s) of an associated URI [p.31] (usually a value of another attribute on the same element). At its most general, it is a comma-separated list of media ranges with optional accept parameters, as defined in section 14.1 of [RFC2616 [p.310]] as the field value of the accept request header. In its simplest case, this is just a media type, such as "image/png" or "application/xml", but it may also contain asterisks, such as "image/*" or "*/*", or lists of acceptable media types, such as "image/png, image/gif, image/jpeg". The user agent must combine this list with its own list of acceptable media types by taking the intersection, and then use the resulting list as the field value of the `accept` request header when requesting the resource using HTTP. For instance, if the attribute specifies the value "image/png, image/gif, image/jpeg", but the user agent does not accept images of type "image/gif" then the resultant accept header would contain "image/png, image/jpeg". A user agent must imitate similar behavior when using other methods than HTTP. For instance, when accessing files in a local filestore, `<p src="logo" srctype="image/png, image/jpeg">` might cause the user agent first to look for a file `logo.png`, and then for `logo.jpg`. If a value for the content type is not given, "*/*" must be used for its value. For the current list of registered content types, please consult [MIMETYPES [p.309]].
Coordinates	Comma separated list of Length [p.31] s used in defining areas.
CURIE	A Compact URI [CURIE [p.309]].
CURIEs	One or more white space separated CURIE [p.30] values
Datetime	Date and time information, as defined by the type `dateTime` in [XMLSCHEMA [p.311]] except that the timezone part is required.
HrefTarget	Name used as destination for results of certain actions, with legal values as defined by NMTOKEN [p.29] .
LanguageCode	A language code. The values should conform to [RFC3066 [p.310]] or its successors.

LanguageCodes	A comma-separated list of language ranges with optional q parameters, as defined in section 14.4 of [RFC2616 [p.310]] as the field value of the Accept-Language request header. Individual language codes should conform to [RFC3066 [p.310]] or its successors.
Length	Either a number, representing a number of pixels, or a percentage, representing a percentage of the available horizontal or vertical space. Thus, the value "50%" means half of the available space.
LocationPath	A location path as defined in [XPATH [p.312]].
MediaDesc	A comma-separated list of media descriptors as described by [CSS2 [p.309]]. The default is `all`.
Number	One or more digits
QName	An [XMLNS [p.311]]-qualified name. See QName for a formal definition.
QNames	One or more white space separated QName [p.31] values
Text	A character string.
URI	An Internationalized Resource Identifier Reference, as defined by [IRI [p.309]].
URIorSafeCURIE	A URI (as defined above) or Safe Compact URI [CURIE [p.309]].
URIorSafeCURIEs	One or more white space separated URIorSafeCURIE [p.31] values.
URIs	A space-separated list of URIs as defined above.

Implementation: RELAX NG [p.196] , XML Schema [p.253]

6. XHTML Attribute Collections

This section is *normative*.

Many of the modules in this document define the required attributes for their elements. The elements in those modules also reference zero or more attribute collections. Attribute collections are defined in their own modules, but the meta collection "Common" is defined in this section. The table below summarizes the attribute collections available.

Note that while these Collections are referenced throughout this document, these expressions should in no way be considered normative or mandatory. They are an editorial convenience. When used in this document, it is the expansion of the term that is normative, not the term itself.

For the avoidance of doubt, if an attribute collection is referenced by an element's definition, and an attribute in that collection is also explicitly referenced by that element's definition, this does NOT cause a conflict. It is up to the schema implementation to update the content models accordingly.

Collection	Module	Description
Core [p.??]	Core Attributes Module [p.61]	Basic attributes used to identify and classify elements and their content.
I18N [p.71]	Internationalization Attribute Module [p.71]	Attribute to identify the language of an element and its contents.
Bi-directional [p.79]	Bi-directional Text Collection [p.79]	Attributes used to manage bi-directional text.
Edit [p.83]	Edit Attributes Module [p.83]	Attributes used to annotate when and how an element's content was edited.
Embedding [p.85]	Embedding Attributes Module [p.85]	Attributes used to embed content from other resources within the current element.
Events [p.147]	XML Events Module [p.147]	Attributes that allow associating of events and event processing with an element and its contents.
Forms [p.144]	XForms Module [p.143]	Attributes that designate provide a mechanism of repeating table rows within a form.
Hypertext [p.65]	Hyptertext Attributes Module [p.65]	Attributes that designate characteristics of links within and among documents.
I18N [p.71]	I18N Attribute Module [p.71]	Attributes for defining the language of the contents of an element.

Collection	Module	Description
Map [p.89]	Image Map Attributes Module [p.89]	Attributes for defining and referencing client-side image maps.
Media [p.93]	Media Attribute Module [p.93]	Attribute for performing element selection based upon media type as defined in MediaDesc [p.31]
Metainformation [p.99]	Metainformation Attributes [p.99]	Attributes that allow associating of elements with metainformation about those elements
Role [p.111]	Role Attribute Module [p.111]	Attribute for the specification of the "role" of an element.
Style [p.119]	Style Attribute Module [p.119]	Attribute for associating style information with an element and its contents.
Common	Attribute Collections Module	A meta-collection of all the other collections, including the Core [p.??] , Bi-directional [p.79] , Events [p.147] , Edit [p.83] , Embedding [p.85] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Media [p.93] , Metainformation [p.99] , Role [p.111] , and Style [p.119] attribute collections.

Implementations: RELAX NG [p.163] , XML Schema [p.243]

7. XHTML Document Module

This section is *normative*.

The Document Module defines the major structural elements for XHTML. These elements effectively act as the basis for the content model of many XHTML family document types. The elements and attributes included in this module are:

Elements	Attributes	Content Model
html [p.35]	Common [p.34] , @version [p.36] (CDATA [p.29]), xmlns (URI [p.31] = "http://www.w3.org/1999/xhtml"), @xsi:schemaLocation [p.36] (URIs [p.31] = "http://www.w3.org/1999/xhtml http://www.w3.org/MarkUp/SCHEMA/xhtml2.xsd")	head [p.36] , body [p.37]
head [p.36]	Common [p.34] , @profile [p.36] (URIs)	title [p.37] , (access [p.73] | action [p.149] | link [p.95] | listener [p.147] | model [p.143] | meta [p.97] | style [p.115]) *
title [p.37]	Common [p.34]	PCDATA*
body [p.37]	Common [p.34]	(Heading [p.39] | Structural [p.39] | List [p.57])*

This module is the basic structural definition for XHTML content. The `html` element acts as the root element for all XHTML Family Document Types.

Note that the value of the xmlns declaration is defined to be "http://www.w3.org/1999/xhtml". Also note that because the xmlns declaration is treated specially by XML namespace-aware parsers [XMLNS [p.311]], it is legal to have it present as an attribute of each element. However, any time the xmlns declaration is used in the context of an XHTML module, whether with a prefix or not, the value of the declaration must be `http://www.w3.org/1999/xhtml`.

Implementations: RELAX NG [p.164] , XML Schema [p.255]

7.1. The html element

The html [p.36] element is the root element for all XHTML Family Document Types. The @xml:lang [p.71] attribute is required on this element.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

version = CDATA [p.29]
: The value of this attribute specifies which XHTML Family document type governs the current document. The format of this attribute value is unspecified. However, all values beginning with the character sequence `xhtml` are reserved for use by XHTML Family Document Types.

xsi:schemaLocation = URIs [p.31]
: This attribute allows the specification of a location where an XML Schema [XMLSCHEMA [p.311]] for the document can be found. The syntax of this attribute is defined in xsi_schemaLocation. The behavior of this attribute in XHTML documents is defined in Strictly Conforming Documents.

7.2. The head element

The head [p.36] element contains information about the current document, such as its title, that is not considered document content. The default presentation of the head is not to display it; however that can be overridden with a style sheet for special purpose use. User agents may however make information in the head [p.36] available to users through other mechanisms.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

profile = URIs [p.31]
: This attribute specifies the location of one or more metadata profiles, separated by white space. For future extensions, user agents should consider the value to be a list even though this specification only considers the first URI to be significant. Profiles are discussed in the section on meta data [p.95] .

Example

```
<head>
        <title>My Life</title>
</head>
```

7.3. The title element

Every XHTML document must have a title [p.37] element in the head [p.36] section.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

The title [p.37] element is used to provide identification for a document or a section of a document. Since documents are often consulted out of context, authors should provide context-rich titles. Thus, instead of a title such as "Introduction", which doesn't provide much contextual background, authors should supply a title such as "Introduction to Medieval Bee-Keeping" instead.

For reasons of accessibility, user agents must always make the content of the title [p.37] element available to users. The mechanism for doing so depends on the user agent (e.g., as a caption, spoken).

When title [p.37] is used in within the head [p.36] element of a document, that title applies to the entire document. When used elsewhere, it applies to its enclosing element.

Example

```
<title>A study of population dynamics</title>
```

Example

```
<section>
    <title>Intoroduction to Medieval Bee-Keeping</title>
    <p>Some information about....</p>
</section>
```

The title of a document is metadata about the document, and so a title like `<title>About W3C</title>` is equivalent to `<meta about="" property="title">About W3C</meta>`.

7.4. The body element

The body of a document contains the document's content. The content may be processed by a user agent in a variety of ways. For example by visual browsers it can be presented as text, images, colors, graphics, etc., an audio user agent may speak the same content, and a search engine may create an index prioritized according to properties of the text.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
<body id="theBody">
    <p>A paragraph</p>
</body>
```

8. XHTML Structural Module

This section is *normative*.

This module defines all of the basic text container elements, attributes, and their content models that are structural in nature.

Element	Attributes	Content Model
address [p.40]	Common [p.34]	(Text [p.47])*
blockcode [p.40]	Common [p.34]	(Text [p.47] | Heading [p.39] | Structural [p.39] | List [p.57])*
blockquote [p.41]	Common [p.34]	(Text [p.47] | Heading [p.39] | Structural [p.39] | List [p.57])*
div [p.41]	Common [p.34]	(Flow [p.39])*
h [p.42]	Common [p.34]	(Text [p.47])*
p [p.43]	Common [p.34]	(Text [p.47] | List [p.57] | blockcode [p.40] | blockquote [p.41] | pre [p.43] | table [p.125])*
pre [p.43]	Common [p.34]	(Text [p.47])*
section [p.44]	Common [p.34]	(Flow [p.39])*
separator [p.45]	Common [p.34]	EMPTY

The content model for this module defines some content sets:

Heading
: h [p.42] | h1 [p.155] | h2 [p.155] | h3 [p.155] | h4 [p.155] | h5 [p.155] | h6 [p.155]

Structural
: action [p.149] | address [p.40] | blockcode [p.40] | blockquote [p.41] | div [p.41] | List [p.57] | listener [p.147] | p [p.43] | pre [p.43] | script [p.151] | section [p.44] | separator [p.45] | style [p.115] | table [p.125]

Flow
: Heading [p.39] | Structural [p.39] | Text [p.47]

Implementations: RELAX NG [p.166] , XML Schema [p.257]

8.1. The address element

The address [p.40] element may be used by authors to supply contact information for a document or a major part of a document such as a form.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
<address href="mailto:webmaster@example.net">Webmaster</address>
```

8.2. The blockcode element

This element indicates that its contents are a block of "code" (see the code [p.49] element). This element is similar to the pre [p.43] element, in that whitespace in the enclosed text has semantic relevance. As a result, the default value of the @layout [p.62] attribute is `relevant`.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example of a code fragment:

```
<blockcode class="Perl">
sub squareFn {
    my $var = shift;
        return $var * $var ;
}
</blockcode>
```

Here is how this might be rendered:

```
sub squareFn {
    my $var = shift;
        return $var * $var ;
}
```

8.3. The blockquote element

This element designates a block of quoted text.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

An excerpt from 'The Two Towers', by J.R.R. Tolkien, as a blockquote

```
<blockquote cite="http://www.example.com/tolkien/twotowers.html">
<p>They went in single file, running like hounds on a strong scent,
and an eager light was in their eyes. Nearly due west the broad
swath of the marching Orcs tramped its ugly slot; the sweet grass
of Rohan had been bruised and blackened as they passed.</p>
</blockquote>
```

8.4. The div element

The div [p.41] element, in conjunction with the @id [p.62] , @class [p.61] and @role [p.111] attributes, offers a generic mechanism for adding extra structure to documents. This element defines no presentational idioms on the content. Thus, authors may use this element in conjunction with style sheets [p.115] , the @xml:lang [p.71] attribute, etc., to tailor XHTML to their own needs and tastes.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

For example, suppose you wish to make a presentation in XHTML, where each slide is enclosed in a separate element. You could use a div [p.41] element, with a @class [p.61] of `slide`:

div with a class of slide

```
<body>
        <h>The meaning of life</h>
        <p>By Huntington B. Snark</p>
        <div class="slide">
                <h>What do I mean by "life"</h>
                <p>....</p>
        </div>
        <div class="slide">
                <h>What do I mean by "mean"?</h>
```

```
                ...
        </div>
        ...
</body>
```

8.5. The h element

The `h` element briefly describes the topic of the section it introduces. Heading information may be used by user agents, for example, to construct a table of contents for a document automatically.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Structured headings use the single h element, in combination with the section [p.44] element to indicate the structure of the document, and the nesting of the sections indicates the importance of the heading. The heading for the section is the one that is a child of the section element.

Example

```
<body>
<h>This is a top level heading</h>
<p>....</p>
<section>
    <p>....</p>
    <h>This is a second-level heading</h>
    <p>....</p>
    <h>This is another second-level heading</h>
    <p>....</p>
</section>
<section>
    <p>....</p>
    <h>This is another second-level heading</h>
    <p>....</p>
    <section>
        <h>This is a third-level heading</h>
        <p>....</p>
    </section>
</section>
</body>
```

Sample style sheet for section levels

```
h {font-family: sans-serif; font-weight: bold; font-size: 200%}
section h {font-size: 150%} /* A second-level heading */
section section h {font-size: 120%} /* A third-level heading */
```

Numbered sections and references
XHTML does not itself cause section numbers to be generated from headings. Style sheet languages such as CSS however allow authors to control the generation of section numbers.

8.6. The p element

The p [p.43] element represents a paragraph.

In comparison with earlier versions of HTML, where a paragraph could only contain inline text, XHTML2's paragraphs represent a richer content model.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Note: We need another example in here contrasting usages.

Example

```
<p>Payment options include:
<ul>
<li>cash</li>
<li>credit card</li>
<li>luncheon vouchers.</li>
</ul>
</p>
```

8.7. The pre element

The pre [p.43] element indicates that whitespace in the enclosed text has semantic relevance. As such, the default value of the @layout [p.62] attribute is relevant.

Note that *all* elements in the XHTML family preserve their whitespace in the document, which is only removed on rendering, via a style sheet, according to the rules of CSS [CSS3-TEXT [p.309]]. This means that in principle any elements may preserve or collapse whitespace on rendering, under control of a style sheet.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

A bad poem where whitespace matters

```
<pre>
                If
             I   had
          any      talent
             I   would
              be a
              poet
</pre>
```

Here is how this might be rendered:

```
                If
             I   had
          any      talent
             I   would
              be a
              poet
```

Note that while historically one use of the pre [p.43] element has been as a container for source code, the new blockcode [p.40] element more appropriate for that.

8.8. The section element

The section [p.44] element, in conjunction with the h [p.42] element, offers a mechanism for structuring documents into sections. This element defines content to be block-level but imposes no other presentational idioms on the content, which may otherwise be controlled from a style sheet.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

By representing the structure of documents explicitly using the section [p.44] and h [p.42] elements gives the author greater control over presentation possibilities than the traditional implicit structuring using numbered levels of headings. For instance, it is then possible to indicate the nesting of sections by causing a border to be displayed to the left of sections.

Example

```
<body>
<h>Events</h>
<section>
        <h>Introduction</h>
        <p>....</p>
        <h>Specifying events</h>
        <p>...</p>
        <section>
```

```
                <h>Attaching events to the handler</h>
                <p>...</p>
        </section>
        <section>
                <h>Attaching events to the listener</h>
                <p>...</p>
        </section>
        <section>
                <h>Specifying the binding elsewhere</h>
                <p>...</p>
        </section>
</section>
</body>
```

8.9. The separator element

The separator [p.45] element separates parts of the document from each other.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
<p>This is some lead in text</p>
<separator />
<p>This is some additional, but separate text.</p>
```

Example

```
<nl>
<label>Navigation</label>
<li href="/">Home</li>
<li><separator/></li>
<li href="prev">Previous</li>
<li href="..">Up</li>
<li href="next">Next</li>
</nl>
```

9. XHTML Text Module

This section is *normative*.

This module defines all of the basic text container elements, attributes, and their content models that are "inline level". Note that while the concept of "inline level" can be construed as a presentation aspect, in this case it is intended to only have a semantic meaning.

Element	Attributes	Content Model
abbr	Common [p.34] , @full [p.48]	(Text [p.47])*
br	Common [p.34]	EMPTY
cite	Common [p.34]	(Text [p.47])*
code	Common [p.34]	(Text [p.47])*
dfn	Common [p.34]	(Text [p.47])*
em	Common [p.34]	(Text [p.47])*
kbd	Common [p.34]	(Text [p.47])*
l	Common [p.34]	(Text [p.47])*
q	Common [p.34]	(Text [p.47])*
samp	Common [p.34]	(Text [p.47])*
span	Common [p.34]	(Text [p.47])*
strong	Common [p.34]	(Text [p.47])*
sub	Common [p.34]	(Text [p.47])*
sup	Common [p.34]	(Text [p.47])*
var	Common [p.34]	(Text [p.47])*

The content model for this module defines a content set:

Text
PCDATA | abbr [p.48] | br [p.48] | cite [p.49] | code [p.49] | del [p.153] | dfn [p.50] | em [p.50] | ins [p.154] | kbd [p.50] | object [p.101] | q [p.51] | ruby [p.113] | samp [p.52] | span [p.52] | strong [p.53] | title [p.37] | var [p.54] | XForms_Form_Controls [p.143]

Implementations: RELAX NG [p.170] , XML Schema [p.260]

9.1. The abbr element

The abbr [p.48] element indicates that a text fragment is an abbreviation (e.g., W3C, XML, Inc., Ltd., Mass., etc.); this includes acronyms.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

full = IDREF [p.29]
: This attribute locates an element that defines the full expansion of an abbreviation. The referenced element must be in the same document as the abbreviation.

The content of the abbr [p.48] element specifies the abbreviated expression itself, as it would normally appear in running text. The @title [p.62] or @full [p.48] attributes may be used to provide the full or expanded form of the expression. Such an attribute should be repeated each time the abbreviation is used in the document.

Examples

```
<abbr title="Limited">Ltd.</abbr>
<abbr title="Abbreviation">abbr.</abbr>
<p>The <dfn id="w3c">World Wide Web Consortium</dfn> (<abbr full="#w3c">W3C</abbr>)
   develops interoperable technologies (specifications, guidelines, software, and tools)
   to lead the Web to its full potential. <abbr full="#w3c">W3C</abbr> is a forum for
   information, commerce, communication, and collective understanding.</p>
```

9.2. The br element

This element is deprecated.

The br [p.48] element indicates that the current output line should be ended at this point, and a new line begun. This element is deprecated in favor of the l [p.51] element.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example:

```
<p class="poem" xml:lang="fr">
Un petit d'un petit<br/>
S'etonne aux Halles.<br/>
Un petit d'un petit,<br/>
Ah! Degres te fallent.
</p>
```

9.3. The cite element

The cite [p.49] element contains a citation or a reference to other sources.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

In the following example, the cite [p.49] element is used to reference the book from which the quotation is taken:

cite as book reference

```
As Gandalf the White said in
<cite cite="http://www.example.com/books/the_two_towers">The Two Towers</cite>,
<quote xml:lang="en">"The hospitality of
your hall is somewhat lessened of late, Theoden King."</quote>
```

cite to reference another specification

```
More information can be found in
<cite cite="http://www.w3.org/TR/REC-xml">[XML]</cite>.
```

9.4. The code element

The code [p.49] element contains a fragment of computer code.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
The Pascal statement <code>i := 1;</code> assigns the
literal value one to the variable <var>i</var>.
```

9.5. The dfn element

The dfn [p.50] element contains the defining instance of the enclosed term.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
<p role="definition">
    An <dfn id="def-acronym">acronym</dfn> is a word
    formed from the initial letters or groups of letters of words in a
    set phrase or series of words.
</p>
```

9.6. The em element

The em [p.50] element indicates emphasis for its contents.

Note that if em [p.50] elements are nested, this indicates a higher level of emphasis.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
Do <em>not</em> phone before 9 a.m.
```

9.7. The kbd element

The kbd [p.50] element indicates input to be entered by the user.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
To exit, type <kbd>QUIT</kbd>.
```

9.8. The l element

The l [p.51] element represents a semantic line of text (e.g., a line of verse or a line of computer code).

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

By retaining structure in text that has to be broken over lines, you retain essential information about its makeup. This gives you greater freedom with styling the content. For instance, line numbers can be generated automatically from the style sheet if needed.

Sample program listing

```
<blockcode class="program">
<l>program p(input, output);</l>
<l>begin</l>
<l>     writeln("Hello world");</l>
<l>end.</l>
</blockcode>
```

CSS Style sheet to number each line

```
.program { counter-reset: linenumber }
l:before {
        position: relative;
        left: -1em;
        counter-increment: linenumber;
        content: counter(linenumber);
}
```

9.9. The q element

This element designates an inline text fragment of quoted text.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Visual user agents must not by default add delimiting quotation marks (as was the case for the `q` element in earlier versions of XHTML and HTML). It is the responsibility of the document author to add any required quotation marks, either directly in the text, or via a style sheet.

Nested quotations using q

```
<p>John said, <q>"I saw Lucy at lunch, she told me
<q>'Mary wants you
to get some ice cream on your way home.'</q> I think I will get
some at Jen and Berry's, on Gloucester Road."</q></p>
```

q with a cite attribute

```
Steven replied:
<q cite="http://lists.example.org/2002/01.html">We quite agree</q>
```

9.10. The samp element

The samp [p.52] element designates sample output from programs, scripts, etc.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
On starting, you will see the prompt <samp>$ </samp>.
```

9.11. The span element

The span [p.52] element, in conjunction with the @id [p.62] , @class [p.61] and @role [p.111] attributes, offers a generic mechanism for adding structure to documents. This element imposes no presentational idioms on the content. Thus, authors may use this element in conjunction with style sheets [p.115] , the @xml:lang [p.71] attribute, the @dir [p.79] attribute etc., to tailor XHTML to their own needs and tastes.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example of span for cross references

```
<p>This operation is called
the <span class="xref">transpose</span>
or <span class="xref">inverse</span>.</p>
```

9.12. The strong element

The strong [p.53] element indicates higher importance for its contents than that of the surrounding content.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
On <strong>Monday</strong> please put the rubbish out,
but <em>not</em> before nightfall!
```

9.13. The sub element

The sub [p.53] element indicates that its contents should regarded as a subscript.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
H<sub>2</sub>O
```

9.14. The sup element

The sup [p.53] element indicates that its contents should be regarded as a super-script.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Many scripts (e.g., French) require superscripts or subscripts for proper rendering. The sub [p.53] and sup [p.53] elements should be used to markup text in these cases.

Example

```
E = mc<sup>2</sup>
<span xml:lang="fr">M<sup>lle</sup> Dupont</span>
```

9.15. The var element

The var [p.54] element indicates an instance of a variable or program argument.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
The parameter <var>ncols</var> represents
the number of colors to use.
```

10. XHTML Hypertext Module

This section is *normative*.

The Hypertext Module provides an element that traditionally has been used in HTML to define hypertext links to other resources. Although all elements may now play the role of a hyperlink (using the @href [p.65] attribute) or hyperlink anchor (using the @id [p.62] attribute), this element remains for explicit markup of links, though is otherwise identical in semantics to the span [p.52] element.

This module supports the following element:

Element	Attributes	Content Model
a [p.55]	Common [p.34]	(Text [p.47])*

Implementations: RELAX NG [p.175] , XML Schema [p.264]

10.1. The a element

An a [p.55] element defines an anchor. Since hypertext attributes such as @href [p.65] may be applied to any element, this element is not strictly necessary, being equivalent to a span [p.52] , but has been retained to allow the expression of explicit links.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Example

```
<a href="http://www.w3.org/">The W3C Home Page</a>
```

11. XHTML List Module

This section is *normative*.

As its name suggests, the List Module provides list-oriented elements. Specifically, the List Module supports the following elements and attributes:

Elements	Attributes	Content Model
dl [p.58]	Common [p.34]	title [p.37] ?, caption [p.81] ?, ((dt [p.58] | dd [p.58])+ | di [p.58] +)
di [p.58]	Common [p.34]	(dt [p.58] +, dd [p.58] *)
dt [p.58]	Common [p.34]	(Text [p.47])*
dd [p.58]	Common [p.34]	(Flow [p.39])*
caption [p.81] , title [p.37] ?,	Common [p.34]	(Text [p.47])*
ul [p.59]	Common [p.34]	title [p.37] ?, caption [p.81] ?, li [p.60] +
li [p.60]	Common [p.34] , @value [p.60]	(Flow [p.39])*

This module also defines the content set List with the content model (dl | ol | ul)+ and adds this set to the Structural [p.39] content set of the Structural [p.39] Module.

Implementations: RELAX NG [p.175] , XML Schema [p.264]

XHTML offers authors several mechanisms for specifying lists of information. Lists may contain:

- Unordered information.
- Ordered information.
- Definitions.

The previous list, for example, is an unordered list, created with the ul [p.59] element:

Example

```
<ul>
<li>Unordered information. </li>
<li>Ordered information. </li>
<li>Definitions. </li>
</ul>
```

An ordered list, created using the ol [p.59] element, contains information where order is important, as in a recipe:

1. Mix dry ingredients thoroughly.
2. Pour in wet ingredients.
3. Mix for 10 minutes.
4. Bake for one hour at 300 degrees.

Definition lists, created using the dl [p.58] element, generally consist of a series of term/definition pairs (although definition lists may have other applications). Thus, when advertising a product, one might use a definition list:

Lower cost
: The new version of this product costs significantly less than the previous one!

Easier to use
: We've changed the product so that it's much easier to use!

Safe for kids
: You can leave your kids alone in a room with this product and they won't get hurt (not a guarantee).

defined in XHTML as:

Example

```
<dl>
<dt>Lower cost</dt>
<dd>The new version of this product costs significantly less than the
previous one!</dd>
<dt>Easier to use</dt>
<dd>We've changed the product so that it's much easier to
use!</dd>
<dt>Safe for kids</dt>
<dd>You can leave your kids alone in a room with this product and
they won't get hurt (not a guarantee).</dd>
</dl>
```

11.1. Definition lists: the dl , di , dt , and dd elements

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Definition lists vary only slightly from other types of lists in that list items consist of two parts: a term and a description. The term is given by the dt [p.58] element. The description is given with a dd [p.58] element. The term and its definition can be grouped within a di [p.58] element to help clarify the relationship between a term and its definition(s).

Example

```
<dl>
  <di>
    <dt>Dweeb</dt>
    <dd>young excitable person who may mature
      into a <em>Nerd</em> or <em>Geek</em></dd>
  </di>
  <di>
    <dt>Hacker</dt>
    <dd>a clever programmer</dd>
  </di>
  <di>
    <dt>Nerd</dt>
    <dd>technically bright but socially inept person</dd>
  </di>
</dl>
```

Here is an example with multiple terms and descriptions:

Example

```
<dl>
   <dt>Center</dt>
   <dt>Centre</dt>
   <dd> A point equidistant from all points
              on the surface of a sphere.</dd>
   <dd> In some field sports, the player who
              holds the middle position on the field, court,
              or forward line.</dd>
</dl>
```

11.2. The ol and ul elements

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Both types of lists are made up of sequences of list items defined by the li [p.60] element. The difference is that ol [p.59] lists represent lists of items that are essentially ordered (such as the steps of a recipe), while ul [p.59] lists represent lists of items that are essentially unordered (such as shopping lists).

Basic list structure

```
<ol>
   <li>Spring</li>
   <li>Summer</li>
   <li>Autumn</li>
   <li>Winter</li>
</ol>
```

11.3. The li element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

value = NUMBER [p.29]
: This attribute specifies the value to be used when determining the value of the enumerator on a list item within an ol [p.59] .

The li [p.60] element defines a list item within an ordered, unordered, or navigation list.

Within a list, each `li` element has an associated number, which is used for numbering list items in ordered lists:

- If the `li` element has a `value` attribute, the associated number is the value of that attribute;
- otherwise, if the `li` element is the first in the list, then the number has the value 1;
- otherwise the number is one higher than the number of the preceding `li` in the same list.

12. XHTML Core Attributes Module

This section is *normative*.

This module defines the Core [p.??] attribute collection.

Attribute	Notes
@class [p.61] (NMTOKENS [p.29])	
@id [p.62] (ID [p.29])	
@xml:id [p.62] (ID [p.29])	
@layout [p.62] (`irrelevant*|relevant`)	
@title [p.62] (Text [p.31])	

12.1. Core Attribute Collection

class = NMTOKENS [p.29]

This attribute assigns one or more class names to an element; the element may be said to belong to these classes. A class name may be shared by several element instances.

The @class [p.61] attribute can be used for different purposes in XHTML, for instance as a style sheet [p.115] selector (when an author wishes to assign style information to a set of elements), and for general purpose processing by user agents.

For instance in the following example, the p [p.43] element is used in conjunction with the @class [p.61] attribute to identify a particular type of paragraph.

Example

```
<p class="note">
These programs are only available if you have purchased
the advanced professional suite.
</p>
```

Style sheet rules can then be used to render the paragraph appropriately, for instance by putting a border around it, giving it a different background color, or where necessary by not displaying it at all.

It is good style to use names that represent the purpose of the element rather than the visual presentation to be used. For instance don't use `class="red"`, but rather `class="urgent"`, or similar.

id = ID [p.29]
The @id [p.62] attribute assigns an identifier to an element. The value of this attribute must be unique within a document. This attribute MUST NOT be specified on an element in conjunction with the @xml:id [p.62] attribute.

The @id [p.62] attribute has several roles in XHTML:

- As a style sheet [p.115] selector.
- As a target anchor [p.55] for hypertext links.
- As the name of a declared object [p.101] element.
- For general purpose processing by user agents (e.g. for identifying fields when extracting data from XHTML pages into a database, translating XHTML documents into other formats, etc.).

As an example, the following headings are distinguished by their @id [p.62] values:

Example

```
<h id="introduction">Introduction</h>
<p>...</p>
<h id="events">The Events Module</h>
<p>...</p>
```

xml:id = ID [p.29]
The @xml:id [p.62] attribute assigns an identifier to an element. The value of this attribute must be unique within a document. This attribute MUST NOT be specified on an element in conjunction with the @id [p.62] attribute.

layout = `irrelevant*|relevant`
This attribute allows authors to indicate whether the whitespace within an element is relevant to the meaning of the content or not; for instance, visual user agents could display the whitespace. The default is that it is *irrelevant*. Some elements, notably pre [p.43] override this default. See whitespace handling in the section on XHTML Family User Agent Conformance for more information.

Example

```
<p class="poem" layout="relevant">
(with wee ears and see?
tail frisks)
              (gonE)
</p>
```

title = Text [p.31]
This attribute defines meta-information about the element on which it is set.

Example

```
<a href="Jakob.html" title="Author biography">Jakob Nielsen</a>'s
Alertbox for January 11, 1998
```

Implementation: RELAX NG [p.178] , XML Schema [p.266]

13. XHTML Hypertext Attributes Module

This section is *normative*.

The Hypertext Attributes Module defines the Hypertext [p.65] attribute collection. This collection allows an element to be the start point of a hypertext link to a remote resource.

13.1. Hypertext Attribute Collection

cite = URI [p.31]

The value of this attribute is a URI [p.31] that designates a source document or message. This attribute is intended to give further information about the element's contents (e.g., the source from which a quotation was borrowed, or the reason text was inserted or deleted). User Agents MUST provide a means for the user to actuate the link.

Example

```
cite="comments.html"
```

href = URI [p.31]

This attribute specifies a URI that is actuated when the element is activated.

Actuation occurs as the default action of a [DOM [p.309]] DOMActivate event for the element on which the attribute occurs (for instance as the result of the user clicking on the associated element). If elements contained within an element using an @href [p.65] also use an @href [p.65] attribute, the User Agent must provide a mechanism for actuating any of these "nested" URIs.

Example

```
<abbr href="http://www.w3.org/" title="World Wide Web">WWW</abbr>
<li href="contents.xhtml">contents</li>
<a href="http://www.cwi.nl/~steven/amsterdam.html">Amsterdam</a>
<quote href="hamlet.xhtml#p2435">To be or not to be</quote>
<var href="#index_ninc">ninc</var>
```

hreflang = LanguageCodes [p.31]

This attribute specifies the primary language of the resource designated by @href [p.65] . At its most general, it is a comma-separated list of language ranges with optional accept parameters, as defined in section 14.4 of [RFC2616 [p.310]] as the field value of the Accept-Language request header.

In its simplest case, this is just a language code, such as "nl", but it may also contain variant specifications such as "en-gb".

The user agent must use this list as the field value of the `accept-language` request header when requesting the resource using HTTP.

If this attribute is not present, the user agent must use its default value of the accept-language request header.

Example

```
<p href="http://www.w3.org/2003/06/semantictour-pressrelease"
   hreflang="fr">
   The press release in French
</p>
```

hrefmedia = MediaDesc [p.31]

This attribute indicates the type(s) of media to which to make available the content referenced by the associated @href [p.65] URI.

Example

```
<p href="http://www.example.com/forPrinters.html"
   hrefmedia="print">
   A printable version of this page.
</p>
```

hreftype = ContentTypes [p.30]

This attribute specifies the allowable content types of the relevant @href [p.65] URI. See ContentTypes [p.30] for details of how it is used.

Example

```
<p href="http://www.w3.org"
   hreftype="text/html,application/xhtml+xml">
   The W3C Home Page
</p>
```

nextfocus = IDREF [p.29]

This attribute specifies an IDREF of an element in the current document that will receive focus when the user requests that the user agent navigate to the next element that can receive focus.

The sequence of focusable [p.18] elements is called the document's navigation order. The navigation order defines the order in which elements will receive focus when navigated by the user. The navigation order may include elements nested within other elements.

When a document is first loaded, a user agent must do the following:

1. If a document is loaded using a URI that includes a reference to a fragment identifier (such as `book.html#chapter5`)
    1. If the fragment reference identifies an element in the document, the user agent must ensure that navigation starts at the beginning of that element.
    2. If the referenced element is focusable, that element receives focus.

3. If the fragment reference does not resolve in the document, the user agent must ensure navigation starts at the beginning of the document.

2. If there is no reference to a fragment identifier when the document is loaded:
   1. If the root element of the document has a @nextfocus [p.66] attribute, and the element referred to by the attribute is focusable, the element must receive focus. The user agent must ensure the beginning of the element is visible on the display.
   2. If the root element has no @nextfocus [p.66] attribute, no element receives initial focus. The user agent must ensure navigation starts at the beginning of the document.
3. If the user has moved away from the initial navigation point of a document (e.g., through using page up and page down or by changing focus), refreshing the document should result in the user's navigation location being preserved.

In the event no element in the document has focus, when the user requests the next focusable element, that element must be the next focusable element forward from the current navigation point in document order. If there are no focusable elements before the end of the document, focus shifts to the first focusable element in document order. If a document has no focusable elements, then no element receives focus.

Once a focusable element in the document has focus, upon requesting that focus change to the next focusable element, the user agent MUST follow these rules when determining where focus is next set:

1. The next focus of an element *without* a @nextfocus [p.66] attribute is the next focusable [p.18] element in document order. If there are no remaining focusable [p.18] elements in document order, the next focus must be on the first focusable [p.18] element in document order.
2. The next focus of an element *with* a @nextfocus [p.66] attribute is the element referenced by that attribute if it is focusable [p.18] , otherwise the next focus of that element.

Regardless of the way in which an element receives focus, if the element is not currently visible on the user agent's display, the display must be updated so that the element is visible.

The following example would allow the links to be navigated in column order (without the use of `nextfocus` they would be navigated in document, i.e. row, order):

Example

```
<table>
<tr><td id="a" href="nw" nextfocus="b">NW</td>
    <td id="c" href="ne" nextfocus="d">NE</td></tr>
<tr><td id="b" href="sw" nextfocus="c">SW</td>
    <td id="d" href="se">SE</td></tr>
</table>
```

Navigation keys. *The actual key sequence that causes navigation or element activation depends on the configuration of the user agent (e.g., the "tab" key might be used for navigation and the "enter" key or "space" key used to activate a selected element).*

prevfocus = IDREF [p.29]
: This attribute specifies an IDREF of an element in the current document that will receive focus when the user requests that user agent navigate to the previous element that can receive focus.

  In the event no element in the document has focus, when the user requests the previous focusable [p.18] element, that element must be the next focusable element backward from the current navigation point in document order. If there is no such focusable element back to the start of the document, focus shifts to the last focusable element in document order. If a document has no focusable elements, the behavior is unspecified.

  Once a focusable element in the document has focus, upon requesting that focus change to the previous focusable element, the user agent must do the following:

  1. If the focused element has a @prevfocus [p.68] attribute that references a focusable element, focus is moved to that element.
  2. Otherwise, focus shifts to the previous focusable element in document order.
  3. If there are no previous focusable elements in document order, focus shifts to the last focusable element in document order.

  Regardless of the way in which an element receives focus, for visual user agents, if the element is not currently visible on the user agent's display, the display must be updated so that the element is visible.

target = HrefTarget [p.30]
: This attribute identifies an environment that will act as the destination for a resource identified by a hyperlink when it is activated.

  This specification does not define how this attribute gets used, since that is defined by the environment that the hyperlink is actuated in. See for instance XFrames [XFRAMES [p.312]]. However, values of this attribute that begin with the character '_' are reserved.

  Example

```
<a href="home.html" target="main">Home</a>
```

xml:base = URI [p.31]
: This attribute specifies the base URI from which to resolve relative URIs. It is normatively defined in [XMLBASE [p.311]]. Any relative URI used on an element that uses this attribute, or on an element contained within an element that uses this attribute, must be resolved relative to the base URI defined by this attribute.

An element inherits URI base information according to the following order of precedence (highest to lowest):

1. The @xml:base [p.68] attribute set for the element itself.
2. The closest parent element that has the @xml:base [p.68] attribute set (i.e., the @xml:base [p.68] attribute is inherited).
3. The HTTP "Content-Location" header (which may be configured in a server).
4. The location of the document itself.

Example

```
See:
<ul xml:base="http://www.w3.org">
<li href="/" src="Icons/w3c_home">The W3C home page</li>
<li href="/TR">The W3C Technical Reports page</li>
<li href="/Markup">The HTML home page</li>
<li href="/Markup/Forms">The XForms home page</li>
</ul>
```

Implementations: RELAX NG [p.180] , XML Schema [p.267]

14. XHTML I18N Attribute Module

This section is *normative*.

This module defines the I18N [p.71] attribute collection.

14.1. I18N Attribute Collection

xml:lang = LanguageCode [p.30]

This attribute indicates the language of an element's attribute values and text content, and of all elements it contains, unless overridden. It is defined normatively in [XML [p.311]] section 2.12. The default value of this attribute is unspecified.

An element inherits language code information according to the following order of precedence (highest to lowest):

1. The @xml:lang [p.71] attribute set for the element itself.
2. The closest parent element that has the @xml:lang [p.71] attribute set (i.e., the @xml:lang [p.71] attribute is inherited).

In this example, the default text-processing language of the document is French ("fr"). Within the first paragraph a single word is declared to be in English ("en"), after which the primary language returns to French. The following paragraph is declared to be in English and includes an embedded French word.

Example

```
<html xmlns="http://www.w3.org/2002/06/xhtml2" xml:lang="fr" ...>
<head>
   <title>Un document multilingue</title>
</head>
<body>
<p>En janvier, toutes les boutiques de Londres
affichent des panneaux <span xml:lang="en">SALE</span>,
mais en fait ces magasins sont bien propres!</p>
<p xml:lang="en">Everwhere I went in France
the bakeries had signs up saying <em xml:lang="fr">PAIN</em>,
but despite that the bakers seemed quite happy.
</p>
</body>
</html>
```

its:translate = ("yes" | "no"*)

This attribute is incorporated from the [ITS [p.309]] specification and can be used to indicate whether the contents of an element should be translated. Note that this element is in the *its* namespace. ITS rules are incorporated into XHTML family documents via the link [p.95] element with a @rel [p.99] attribute value of `itsRules`.

Implementations: RELAX NG [p.181] , XML Schema [p.268]

15. XHTML Access Module

This section is *normative* for purposes of defining the integration of the XHTML Access Module into XHTML 2. The semantics of the XHTML Access Module itself are normatively defined in [XHTMLACCESS [p.311]].

This module defines the access [p.73] element.

Element	Attributes	Content Model
access [p.73]	Common [p.34] , @activate [p.74] , @key [p.74] , @media [p.75] , @order [p.75] , @targetid [p.76] , @targetrole [p.76]	EMPTY

Implementations: RELAX NG [p.182] , XML Schema [p.282]

The access [p.73] element assigns a mapping between "keys" or other events to elements within a document. Actuating the mapping results in the element gaining focus and potentially in additional events being activated.

15.1. The access element

The access [p.73] element assigns a mapping between "keys" or other events to elements within a document. Actuating the mapping results in the element gaining focus and potentially in additional events being activated.

An access element must have either a targetrole or a targetid attribute specified. If neither a targetrole nor a targetid attribute are specified, the user agent MUST NOT define a mapping nor dispatch any events.

The access [p.73] element allows an author to specify a relationship between "key(s)" or other events and one or more elements within a document. When a mapped event is triggered, one of the specified target elements gains focus, and one or more other events (such as an 'activate' event) may also be dispatched to that element. The target elements are specified by means of the @targetid [p.76] or @targetrole [p.76] attributes, and these elements may each contain multiple targets, to allow sequential focus by successively triggering the associated key(s) or event(s).

If the access [p.73] element's @activate [p.74] attribute has the value 'true', then in addition to receiving focus, the target element will be activated, if permitted by the user's settings. Additionally, using XML Events [XMLEVENTS [p.311]], one or more other events may also be dispatched to the element.

An access element must have either a @targetrole [p.76] or a @targetid [p.76] attribute specified. If neither a @targetrole [p.76] nor a @targetid [p.76] attribute are specified, or if there are no matching elements in the document, the user agent MUST NOT define a mapping for this

element, nor change focus nor dispatch any events based on this element.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

activate = ("true" | "false"*)
: The @activate [p.74] attribute indicates whether a target element should be activated or not once it obtains focus. The default value for this attribute is "false", indicating that the element will not be "activated". User agents MUST provide mechanisms for overriding the author setting with user-specified settings in order to ensure that the act of moving content focus does not cause the user agent to take any further action (as per Checkpoint 9.5 of UAAG 1.0 [UAAG1 [p.310]]).

  User agents MUST provide keyboard mechanisms for "activating" any event associated with the focused element (as per Checkpoint 1.2 of UAAG 1.0). User agents SHOULD make available the list of events associated with the focused element (as per Checkpoint 9.6 of UAAG 1.0).

key = Characters [p.29]
: This attribute assigns one or more key mappings to an access shortcut. The value of is attribute is one or more single characters from the document character set.

  The @key [p.74] attribute represents an abstraction. The use of the name "key" for this attribute is historical and does not mean that there is any association with a specific "key" on a keyboard, per se. It is up to the user agent to provide a mechanism for mapping the document character set value(s) of the attribute to the input methods available to the user agent. For instance, on some systems a user may have to press an "alt" or "cmd" key in addition to the access key. On other systems there may be voice commands, or gestures with a mouse, an eye tracking input device, etc. Regardless of the mechanism, the result is that it appears to the `access` element processor as if the defined *key* was entered.

  A user entering any of the keys defined in an access element moves focus from its current position to the next element in navigation order that has one of the referenced role or id values (see @activate [p.74] for information on how the element may be activated). Note that it is possible to deliver alternate events via [XMLEVENTS [p.311]]. Note also that the concept of *navigation order* is a property of the Host Language, and is not defined in this specification.

  User agents MUST provide mechanisms for overriding the author setting with user-specified settings in order to ensure that the act of moving content focus does not cause the user agent to take any further action, as required by UAAG 1.0, Checkpoint 9.5. [UAAG1 [p.310]] The character assigned to a key, and its relationship to a role or id attribute SHOULD be treated as an author suggestion. User agents MAY override any key assignment (e.g., if an assignment interferes with the operation of the user interface of the user agent, if the key is

not available on a device, if a key is used by the operating environment). User agents MUST also allow users to override author assigned keys with their own key assignments (see Checkpoint 11.3 of UAAG 1.0). If a user chooses to change the key binding, the resultant user-defined remapping SHOULD persist across sessions.

If no @key [p.74] attribute is specified, the user agent SHOULD assign a key and alert the user to the key mapping. The resultant user agent assigned key mapping SHOULD persist across sessions.

It is common for user agents to provide a visual hint for accessing features via keyboard shortcuts, such as showing the shortcut next to a menu item, or underlining the shortcut character letter in a label. The rendering of such visual hints about access keys depends on the user agent. We recommend that authors include the access key character in label text or wherever the access key is to apply. If the user agent can recognize that the currently mapped access key character appears in the label text of the element to which it is mapped, then the user agent may render the character in such a way as to emphasize its role as the access key and distinguish it from other characters (e.g., by underlining it).

A conforming user agent SHOULD also provide a centralized view of the current access key assignments (see Checkpoint 11.1 and Checkpoint 11.2 of UAAG 1.0).

Authors MUST NOT assign the same @key [p.74] value to more than one *active* `access` element. Note that it is possible to have an `access` element be *inactive* through the use of the @media [p.75] attribute.

media = MediaDesc [p.31]

The value of this attribute is a comma-separated list of media descriptors for which this `access` element is intended. When the value of this attribute matches the current processing media, the associated `access` element is considered *active* and processed normally; otherwise it is *inactive* and ignored. The default value for this attribute is `all`.

Example

```
<style src="style.css" type="text/css" media="screen" />
<span src="photo.jpg" media="screen">Me at work</span>
<span src="photo-hires.jpg" media="print">Me at work</span>
```

order = ("document"* | "list")

The @order [p.75] attribute indicates how this `access` element should determine the *navigation order* for its *matching elements*. The default value, `document`, indicates that the items MUST be traversed in document order. The alternate value, `list`, indicates that the *navigation order* of *matching elements* is determined by the author-defined order of items in the list of @targetid [p.76] or @targetrole [p.76] attribute values.

For the sake of determining *navigation order*, elements in the document that match the values in the @targetid [p.76] or @targetrole [p.76] attributes are called *matching elements*, and all elements that match the same value are members of an *element group*. Note: since the `id` of an element must be unique within a valid XML document, in such documents, each *element group* based on @targetid [p.76] values consist of no more than one

matching element.

For each navigation operation, the *navigation order* for both *document-order* and *list-order* is sensitive to the current *focus* element. For document-order, if no element currently has *focus*, the first *matching element* in the document MUST be the *focus target*; if an element does have *focus*, the next *matching element* in the document MUST be the *focus target*. For list-order, the *focus target navigation order* is determined as follows:

- If no *matching element* of this `access` element currently has *focus*, the *focus target* MUST be the first element in document order that matches the first *element group*. If there is no such element, then the *focus target* is the first element that matches the second *element group*, and so on.
- If a *matching element* of this `access` element already has *focus*:
  1. If there are additional *matching elements* of the same *element group* in document order, then *focus* MUST be sent to the next *matching element* of the same *element group*.
  2. Otherwise, *focus* MUST go to the first *matching element* (in document order) of the next *element group*.
  3. If there are no remaining elements groups, then the search MUST resume from the first *element group*.

The location of the next *matching element* MUST be determined each time the `access` element is triggered, since it is possible that between events the contents of the document will have changed.

targetid = IDREF [p.29]
: The @targetid [p.76] attribute specifies one or more space separated IDREF [p.29] s related to target elements for the associated event (i.e., the node to which the event should be delivered).

targetrole = CURIEs [p.30]
: The @targetrole [p.76] attribute specifies space separated list of CURIE [p.30] s that maps to an element with a @role [p.111] attribute with the same value.

15.1.1. Examples

Access element that focuses into a field

```
<access key="s"
        title="Social Security Number"
        targetrole="ss:number" />
```

Accessing a table of contents

```
<access key="c"
        title="Table of Contents"
        targetrole="toc" />
```

Access that moves to the main content

```
<access key="m"
        title="Main content"
        targetrole="main" />
```

Access that moves among form controls in document order

```
<access key="i"
        title="Form Controls"
        order="document"
        targetrole="button checkbox option radio textbox" />
```

Access that moves among form controls in specific order

```
<access key="i"
        title="Form Controls"
        order="list"
        targetrole="button checkbox option radio textbox" />
```

Access element that goes to a specific element

```
<access key="u"
        title="Username"
                targetid="username" />
```

Access element with no specific key mapping

```
<access title="Navigation bar"
                targetrole="navigation" />
```

16. XHTML Bi-directional Text Attribute Module

This section is *normative*.

The Bi-directional Text module defines the Bi-directional [p.79] attribute collection.

16.1. Bi-directional Text Collection

dir = "`ltr`|`rtl`|`lro`|`rlo`"

This attribute allows the author to specify the direction of the element's text content. This direction affects the display of characters as defined in Unicode Standard Annex #9: The Bidirectional Algorithm [UAX9 [p.310]], and defines directional properties of text as defined by CSS2 [CSS2 [p.309]]. The default value of this attribute is `ltr`. Possible values are:

- `ltr`: Left-to-right text. The effect of this attribute is defined by the CSS2 rule:

```
*[dir="ltr"] { unicode-bidi: embed; direction: ltr}
```

- `rtl`: Right-to-left text. The effect of this attribute is defined by the CSS2 rule:

```
*[dir="rtl"] { unicode-bidi: embed; direction: rtl}
```

- `lro`: Left-to-right override. The effect of this attribute is defined by the CSS2 rule:

```
*[dir="lro"] { unicode-bidi: bidi-override; direction: ltr}
```

- `rlo`: Right-to-left override. The effect of this attribute is defined by the CSS2 rule:

```
*[dir="rlo"] { unicode-bidi: bidi-override; direction: rtl}
```

Example

```
<p>
The Hebrew word for "Hebrew" is
<span xml:lang="he">&#x5e2;&#x5d1;&#x5e8;&#x5d9;&#x5ea;</span>,
but since Hebrew letters have intrinsic right-to-left directionality,
I had to type the word starting from the letter "&#x5e2;",
i.e. <span xml:lang="he" dir="lro">&#x5e2;&#x5d1;&#x5e8;&#x5d9;&#x5ea;</span>.
</p>
```

This might display as

The Hebrew word for "Hebrew" is עברית, but since Hebrew letters have intrinsic right-to-left directionality, I had to type the word starting from the letter "ע", i.e. תירבע.

16.1.1. Inheritance of text direction information

The Unicode bidirectional algorithm requires a base text direction for text blocks. To specify the base direction of a block-level element, set the element's @dir [p.79] attribute. The default value of the @dir [p.79] attribute is `ltr` (left-to-right text).

When the @dir [p.79] attribute is set for a block-level element, it remains in effect for the duration of the element and any nested block-level elements. Setting the @dir [p.79] attribute on a nested element overrides the inherited value.

To set the base text direction for an entire document, set the @dir [p.79] attribute on the html [p.36] element.

Example

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 2.0//EN"
   "<dtdloc>">
<html xmlns="<xmlns>" dir="rtl">
<head>
<title><em>...a right-to-left title...</em></title>
</head>
<body>
<em>...right-to-left text...</em>
<p dir="ltr"><em>...left-to-right text...</em></p>
<p><em>...right-to-left text again...</em></p>
</body>
</html>
```

Inline-level elements, on the other hand, do not inherit the @dir [p.79] attribute. This means that an inline element without a @dir [p.79] attribute does **not** open an additional level of embedding with respect to the bidirectional algorithm.

An element is considered to be block-level if its presentation, when expressed in [CSS2 [p.309]], is `display: block` and inline-level if its presentation, when expressed in [CSS2 [p.309]], is `display: inline`.

16.1.2. The effect of style sheets on bidirectionality

In general, using style sheets (such as [CSS2 [p.309]] to change an element's visual rendering from the equivalent of `display: block` to `display: inline` or vice-versa is straightforward. However, because the bidirectional algorithm relies on the inline/block-level distinction, special care must be taken during the transformation.

When an inline-level element that does not have a @dir [p.79] attribute is transformed to a block-level element by a style sheet, it inherits the @dir [p.79] attribute from its closest parent block-level element to define the base direction of the block.

When a block-level element that does not have a @dir [p.79] attribute is transformed to an inline-level element by a style sheet, the resulting presentation should be equivalent, in terms of bidirectional formatting, to the formatting obtained by explicitly adding a @dir [p.79] attribute (assigned the inherited value) to the transformed element.

Implementations: RELAX NG [p.183] , XML Schema [p.269]

17. XHTML Caption Module

This section is *normative*.

The Caption Module defines an element to be used when annotating a description for certain elements. The element and attributes defined by this module are:

Elements	Attributes	Content Model
caption [p.81]	Common [p.34]	(Text [p.47])*

Implementations: RELAX NG [p.202] , XML Schema [p.269]

17.1. The caption element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

When present, the caption [p.81] element"s text should describe the nature of the table, list or object. The caption [p.81] element is only permitted immediately after the table [p.125] start tag, the ol [p.59] start tag or the object [p.101] start tag. A table [p.125] , ol [p.59] or object [p.101] element may only contain one caption [p.81] element.

For tables, visual user agents allow sighted people to quickly grasp the structure of the table from the headings as well as the caption. A consequence of this is that captions will often be inadequate as a summary of the purpose and structure of the table from the perspective of people relying on non-visual user agents.

18. XHTML Edit Attributes Module

This section is *normative*.

This module defines the Edit [p.83] attribute collection.

18.1. Edit Collection

edit = "`inserted`|`deleted`|`changed`|`moved`"

This attribute allows elements to carry information indicating how content has changed. Possible values:

- `inserted`: the content has been inserted
- `deleted`: the content has been deleted
- `changed`: the content has changed considerably, therefore making it not worth being marked up with values of `inserted` and `deleted`
- `moved`: the content has been moved from some other part of the document.

Example

```
<p>I will do it
next <span edit="deleted">week</span><span edit="inserted">month</span>.</p>
```

datetime = Datetime [p.30]

The value of this attribute specifies the date and time when a change was made.

Example

```
datetime="2003-01-13T13:15:30Z"
```

Implementations: RELAX NG [p.183] , XML Schema [p.270]

19. XHTML Embedding Attributes Module

This section is *normative*.

The Embedding Attributes module defines the Embedding [p.85] attribute collection.

This collection causes the contents of a remote resource to be embedded in the document in place of the element's content. If accessing the remote resource fails, for whatever reason (e.g., network unavailable, no resource available at the URI given, inability of the user agent to process the type of resource) or an associated @ismap [p.90] attribute fails, the content of the element must be processed instead.

Note that this behavior makes documents far more robust, and gives much better opportunities for accessible documents than the `longdesc` attribute present in earlier versions of XHTML, since it allows the description of the resource to be included in the document itself, rather than in a separate document.

Example

```
<p src="holiday.png" srctype="image/png">
        <span src="holiday.gif" srctype="image/gif">
                An image of us on holiday.
        </span>
</p>
<table src="temperature-graph.png" srctype="image/png">
<caption>Average monthly temperature over the last 20 years</caption>
<tr><th>Jan</th><th>Feb</th><th>Mar</th><th>Apr</th><th>May</th><th>Jun</th>
    <th>Jul</th><th>Aug</th><th>Sep</th><th>Oct</th><th>Nov</th><th>Dec</th>
</tr>
<tr><td> 4</td><td> 2</td><td> 7</td><td> 9</td><td>13</td><td>16</td>
    <td>17</td><td>17</td><td>14</td><td>11</td><td> 7</td><td> 4</td>
</tr>
</table>
```

19.1. Embedding Attribute Collection

encoding = Encodings [p.29]

This attribute specifies the allowable encoding of the external resource referenced by the @src [p.86] attribute. At its most general, it is a comma-separated list of encodings, such as "utf-8", "utf8, utf-16", or "utf-8, utf-16, *".

The user agent must use this list as the field value of the `accept-charset` request header when requesting the resource using HTTP.

If this attribute is not present, the user agent must use its default value of the accept-charset request header.

User agents should use a similar technique when using other protocols that allow encoding negotation

When using protocols that do not allow encoding negotiation to retrieve resources whose encodings are not self-identifying, the user agent should use the first encoding in the attribute's value as the indication of the resource.

Example

```
<style type="text/css" src="style/home" encoding="utf-8" />
```

src = URI [p.31]

This attribute specifies the location of an external source for the contents of the element. Actuation occurs as the default action of a [DOM [p.309]] load event for the element that the attribute occurs on.

srctype = ContentTypes [p.30]

This attribute specifies the allowable content types of the resource referenced by the relevant @src [p.86] URI.

Example

```
<handler src="pop" srctype="application/x-javascript, text/x-newspeak" />
<style src="midnight" srctype="text/css, text/x-mystyle" />
<p src="w3c-logo" srctype="image/png, image/jpeg;q=0.2">W3C logo</p>
<span src="logo.png">Our logo</span>
<span src="theme.mp3" srctype="audio/x-mpeg">Our theme jingle</span>
```

Implementations: RELAX NG [p.184] , XML Schema [p.270]

20. XHTML Image Module

This section is *normative*.

The Image Module provides basic image embedding, and may be used in some implementations independently of client side image maps. The Image Module supports the following element and attributes:

Elements	Attributes	Content Model
img	Common [p.34]	(caption [p.81] | Text [p.47])*

Implementations: RELAX NG [p.185] , XML Schema [p.271]

20.1. The img element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

The img [p.87] element is a holder for embedding attributes such as @src [p.86] . Since these attributes may be applied to any element, the img [p.87] element is not strictly necessary, but is included to ease the transition to XHTML2. Like the object [p.101] element, this element's content is only presented if the referenced resource is unavailable.

In the following example, the W3C logo would be presented if it were available. If it were unavailable, then the enclosed text would be presented.

Example

```
<img alt="W3C Logo" src="W3C.png">W3C</img>
```

21. XHTML Image Map Attributes Module

This section is *normative*.

This module defines a collection of attributes that specify that an embedded image may be used as an image map, so that clicking on different parts of the image causes different hyperlinks to be activated.

Note that in the following example, if the image is unavailable for any reason, the fallback properties of the @src [p.86] attribute mean that the `nl` element will be displayed instead of the image, thus making the page still useful:

Example

```
<html xmlns="http://www.w3.org/2002/06/xhtml2">
   <head>
      <title>The cool site!</title>
   </head>
   <body>
     <ul id="map1" role="navigation"
              src="navbar1.png" srctype="image/png" usemap="#map1">
           <caption>Navigate the site:</caption>
           <li href="guide.html" shape="rect"
                   coords="0,0,118,28">
                   Access Guide
           </li>
           <li href="shortcut.html" shape="rect"
                   coords="118,0,184,28">
                   Go
           </li>
           <li href="search.html" shape="circle"
                   coords="184,200,60">
                   Search
           </li>
           <li href="top10.html" shape="poly"
                   coords="276,0,276,28,100,200,50,50,276,0">
                   Top Ten
           </li>
         </ul>
   </body>
</html>
```

Note that W3C is working on profiles of XHTML that include versions of SVG [SVG [p.312]], which include more structured ways of creating imagemap-like behavior.

21.1. Image Map Attribute Collection

usemap = URI [p.31]

This attribute associates an image map with an ul [p.59] element. The value of `usemap` should match the value of the @id [p.62] attribute of an ul [p.59] element that contains one or more li [p.60] elements with @shape [p.90] and @coords [p.90] attributes.

If accessing the URI fails or the referenced element is not an ul [p.59] element, then the associated @src [p.86] attribute is considered to have failed as well, so that the nested content will be processed instead.

ismap = "ismap"

This attribute indicates that the associated embedded resource is to be treated as a "server-side image map". When selected, the coordinates within the element that the user selected are sent to the server where the document resides. Coordinates are expressed as pixel values relative to the embedded resource, and start at (0,0) at the top left corner.

When an ismap attribute is specified, click events are not delivered to the embedded resource, regardless of its type.

In the following example, the active region defines a server-side image map. A click anywhere on the image will cause the click's coordinates to be sent to the server.

```
<p href="http://www.example.com/cgi-bin/map"
      src="map.png" ismap="ismap">
   Our location.
</p>
```

The location clicked is passed to the server as follows. The user agent derives a new URI from the URI specified by the @href [p.65] attribute of the element, by appending '?' followed by the x and y coordinates, separated by a comma. The link is then actuated using the new URI. For instance, in the given example, if the user clicks at the location x=10, y=27 then the derived URI is "http://www.example.com/cgi-bin/map?10,27".

User agents that do not offer the user a means to select specific coordinates (e.g., non-graphical user agents that rely on keyboard input, speech-based user agents, etc.) must send the coordinates "0,0" to the server when the link is activated.

shape = "`default`|`rect`|`circle`|`poly`"

This attribute specifies the shape of a region. Possible values:

- `default`: Specifies the entire region.
- `rect`: Define a rectangular region.
- `circle`: Define a circular region.
- `poly`: Define a polygonal region.

coords = Coordinates [p.30]

This attribute specifies the position and shape of the area. The number and order of values depends on the value of the @shape [p.90] attribute. Possible combinations:

- `rect`: left-x, top-y, right-x, bottom-y.
- `circle`: center-x, center-y, radius. When the radius value is a percentage, the actual

radius value is calculated using the associated image's width and height. The radius is then the smaller value of the two.

- `poly`: x1, y1, x2, y2, ..., xN, yN. If the first and last x and y coordinate pairs are not the same, user agents must infer an additional coordinate pair to close the polygon.

Coordinates are relative to the top, left corner of the object. All values are of type Length [p.31] . All values are separated by commas. The coordinates of the top, left corner of an area are `0, 0`.

Implementation: RELAX NG [p.185] , XML Schema [p.272]

22. XHTML Media Attribute Module

This section is *normative*.

The Media Attribute Module defines the `media` attribute.

22.1. Media Attribute Collection

media = MediaDesc [p.31]
: The value of this attribute is a comma-separated list of media descriptors for which this `access` element is intended. When the value of this attribute matches the current processing media, the associated `access` element is considered *active* and processed normally; otherwise it is *inactive* and ignored. The default value for this attribute is `all`.

Implementations: RELAX NG [p.186] , XML Schema [p.272]

23. XHTML Metainformation Module

This section is *normative*.

The Metainformation Module defines elements that allow the definition of relationships. These may relate to:

- the document itself,
- items external to the document, or
- other items of metadata within the document.

Note that this module is dependent upon the Metainformation Attributes [p.99] module. The interpretation of those attributes in conjunction with any elements, including the ones defined in this module, are spelled out in [RDFASYNTAX [p.310]].

Elements and attributes in this module are:

Elements	Attributes	Content Model
link	Common [p.34]	EMPTY
meta	Common [p.34]	meta [p.97] ?

Implementations: RELAX NG [p.188] , XML Schema [p.273]

23.1. The link element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

This element defines a link. Link [p.95] conveys relationship information that may be rendered by user agents in a variety of ways (e.g., a tool-bar with a drop-down menu of links). User agents should enable activation of links and the retrieval of link targets. Since link [p.95] elements may have no content, information from the @rel [p.99] and @title [p.62] attributes should be used when labelling links.

This example illustrates how several link [p.95] definitions may appear in the head [p.36] section of a document. The current document is "Chapter2.html". The @rel [p.99] attribute specifies the relationship of the linked document with the current document. The values "Index", "Next", and "Prev" are explained in the section on the attribute @rel [p.99] .

```
<head>
  <title>Chapter 2</title>
  <link rel="index" href="../index.html"/>
  <link rel="next"  href="Chapter3.html"/>
  <link rel="prev"  href="Chapter1.html"/>
</head>
```

23.1.1. Forward and reverse links

While the @rel [p.99] attribute specifies a relationship *from* this document *to* another resource, the @rev [p.99] attribute specifies the reverse relationship.

Consider two documents A and B.

```
Document A:       <link href="docB" rel="index"/>
```

Has exactly the same meaning as:

```
Document B:       <link href="docA" rev="index"/>
```

namely that document B is the index for document A.

Both the @rel [p.99] and @rev [p.99] attributes may be specified simultaneously.

23.1.2. Links and search engines

Authors may use the link [p.95] element to provide a variety of information to search engines, including:

- Links to alternate versions of a document, written in another human language.
- Links to alternate versions of a document, designed for different media, for instance a version especially suited for printing.
- Links to the starting page of a collection of documents.

The examples below illustrate how language information, media types, and link types may be combined to improve document handling by search engines.

The following example shows how to use the @hreflang [p.65] attribute to indicate to a search engine where to find other language versions of a document. Note that for the sake of the example the @xml:lang [p.71] attribute has been used to indicate that the value of the @title [p.62] attribute for the link [p.95] element designating the French manual is in French.

```
<html ... xml:lang="en">
<head>
<title>The manual in English</title>
<link title="The manual in Dutch"
      rel="alternate"
      hreflang="nl"
      href="http://example.com/manual/dutch.html"/>
<link title="La documentation en Français"
```

```
      rel="alternate"
      hreflang="fr" xml:lang="fr"
      href="http://example.com/manual/french.html"/>
</head>
```

In the following example, we tell search engines where to find the printed version of a manual.

```
<head>
<title>Reference manual</title>
<link media="print"
      title="The manual in PostScript"
      hreftype="application/postscript"
      rel="alternate"
      href="http://example.com/manual/postscript.ps"/>
</head>
```

In the following example, we tell search engines where to find the front page of a collection of documents.

```
<head>
<title>Reference manual -- Chapter 5</title>
<link rel="start" title="The first chapter of the manual"
      hreftype="application/xhtml+xml"
      href="http://example.com/manual/start.html"/>
</head>
```

23.2. The meta element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

The meta [p.97] element can be used to identify properties of a document (e.g., author, expiration date, a list of key words, etc.) and assign values to those properties. This specification defines a small normative set of properties, but users may extend this set as described for the @property [p.99] attribute.

Each meta [p.97] element specifies a property/value pair. The @property [p.99] attribute identifies the property and the content of the element or the value of the @content [p.99] attribute specifies the property's value.

For example, the following declaration sets a value for the `Author` property:

Example

```
<meta property="dc:creator">Steven Pemberton</meta>
```

Note. *The meta [p.97] element is a generic mechanism for specifying metadata. However, some XHTML elements and attributes already handle certain pieces of metadata and may be used by authors instead of meta [p.97] to specify those pieces: the title [p.37] element, the address [p.40] element, the @edit [p.83] and related attributes, the @title [p.62] attribute, and the @cite [p.65] attribute.*

Note. *When a property specified by a meta [p.97] element takes a value that is a URI [p.31] , some authors prefer to specify the metadata via the link [p.95] element. Thus, the following metadata declaration:*

Example

```
<meta property="dc:identifier">
      http://www.rfc-editor.org/rfc/rfc3236.txt
</meta>
```

might also be written:

Example

```
<link rel="dc:identifier"
      href="http://www.rfc-editor.org/rfc/rfc3236.txt" />
```

23.2.1. meta and search engines

A common use for meta [p.97] is to specify keywords that a search engine may use to improve the quality of search results. When several meta [p.97] elements provide language-dependent information about a document, search engines may filter on the @xml:lang [p.71] attribute to display search results using the language preferences of the user. For example,

Example

```
<!-- For speakers of US English -->
<meta property="keywords"
      xml:lang="en-us">vacation, Greece, sunshine</meta>
<!-- For speakers of British English -->
<meta property="keywords"
      xml:lang="en">holiday, Greece, sunshine</meta>
<!-- For speakers of French -->
<meta property="keywords"
      xml:lang="fr">vacances, Grèce, soleil</meta>
```

The effectiveness of search engines can also be increased by using the link [p.95] element to specify links to translations of the document in other languages, links to versions of the document in other media (e.g., PDF), and, when the document is part of a collection, links to an appropriate starting point for browsing the collection.

24. XHTML Metainformation Attributes Module

This section is *normative* for purposes of defining the integration of the XHTML Metainformation Attributes Module into XHTML 2. The semantics of the XHTML Metainformation Attributes Module itself are normatively defined in [RDFASYNTAX [p.310]]. The rules for extracting RDF from XHTML family markup languages are defined in [RDFASYNTAX [p.310]]. For information on important differences between XHTML 2 and other XHTML family markup languages and how those may relate to RDFa, see Appendix A [p.157] .

24.1. Metadata Attribute Collection

Attributes	Notes
about (URIorSafeCURIE [p.31])	
content (CDATA [p.29])	
datatype (CURIE [p.30])	If not specified, then the default value is `string` as defined in [XMLSCHEMA [p.311]].
typeof (CURIEs [p.30])	
property (CURIEs [p.30])	
rel (reserved word | CURIE [p.30])+	See the reserved values list in [XHTMLVOCAB [p.311]]
resource (URIorSafeCURIE [p.31])	
rev (reserved word | CURIE [p.30])+	See the reserved values list in [XHTMLVOCAB [p.311]]

Implementations: RelaxNG [p.186] , XML Schema [p.274]

25. XHTML Object Module

This section is *normative*.

The Object Module provides elements for general-purpose object inclusion; this includes images and other media, as well as executable content. Specifically, the Object Module supports:

Elements	Attributes	Content Model
object	Common [p.34] , @archive [p.101] (URIs [p.31]), @content-length [p.101] (Number [p.31]), @declare [p.101] ("declare")	(caption [p.81] ?, title [p.37] ?, standby [p.110] ?, param [p.106] *, (Flow [p.39])*)
param	@id [p.62] (ID [p.29]), @name [p.106] * (CDATA [p.29]), @value [p.106] (CDATA [p.29]), @valuetype [p.106] ("data"* | "ref" | "object"), @paramtype [p.106] (ContentType [p.29])	EMPTY
standby	Common [p.34]	(Text [p.47])*

Implementations: RELAX NG [p.189] , XML Schema [p.274]

25.1. The object element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

archive = URIs [p.31]
: This attribute may be used to specify a space-separated list of URIs for archives containing resources relevant to the object, which may include the resources specified by the @src [p.86] attribute.

content-length = Number [p.31]
: This attribute is to be used as a hint by the object handler. The author may provide the object handler with the physical size of the object data that is to be processed. A valid value is the same as defined in section 14.13 of [RFC2616 [p.310]].

declare = "`declare`"
: When present, this boolean attribute makes the current element a declaration only - one that is to be executed only after the document has completed loading and has been called through a user event.

25.1.1. Defining terminology

The following terms are used throughout this section.

object src (source)
: The file that is to be processed, such as an audio file, or image file. The actual content to be processed.

object handler
: The mechanism that will be used to process the object data. The mechanism could be the user agent or an external application.

object element
: This refers to the actual XHTML coding, including the allowable attributes.

instantiation
: Refers to the plug-in handler, and the need to create a window, modify the user interface, allocate memory, etc.

25.1.2. Basic Information for Object Handlers

Most user agents have built-in mechanisms for processing common data types such as text, and various image types. In some instances the user agent may pass the processing to an external application. Generally, a user agent will attempt to process the object declaration, otherwise it may invoke an external application, which are normally referred to as "plug-ins".

In the most general case, an author should specify three types of information:

- The location of the object data (the src attribute). The author must direct the object handler to the actual physical location of the object data, otherwise the object handler will not be able to process the request.
- The media type associated with the object data (the type attribute). For instance, if the author prefers that a particular object handler be used to process the data, they may specify a media type that is associated to a specific object handler.
- Additional values required for the appropriate processing of the object data by the object handler at run-time (via the param element). Some instances may process more appropriately if the object handler is passed initial process instructions. For example, the author can specify whether a video should automatically start or wait until the entire data file has been downloaded.

The object [p.101] element allows authors to specify all three types of information, but authors may not have to specify all three at once. For example, some object element instances may not require src (e.g., a self-contained applet that performs a small animation). Others may not require media type information, i.e., the user agent itself may already know how to process that type of data. Still others may not require run-time initialization.

The object [p.101] element may also appear in the content of the head [p.36] element. Since user agents generally do not render elements in the head [p.36] , authors should ensure that any object [p.101] element in the head [p.36] does not specify content that is expected to be made available to the user.

25.1.3. Rules for processing objects

A user agent must interpret an object [p.101] element according to the following precedence rules:

1. The user agent MUST first try to process the object element. It should not process the embedded contents, but it must examine them for definitions of param [p.106] elements (see object initialization) or elements that take advantage of the Map [p.89] attribute collection.
2. If the user agent is not able to process the object for whatever reason (configured not to, lack of resources, wrong architecture, etc.), it MUST try to process its contents.

When a user agent is able to successfully process an object element it MUST not attempt to process inner elements.

If a user agent cannot process an object element or a set of nested objects, and the author did not provide alternate text, the user agent SHOULD NOT supply any additional information. It is the responsibility of the author to supply additional or alternate information. It may be the intent of the author to not provide additional information if the object cannot be processed.

The user agent SHOULD attempt to process the outer object to its fullest extent before cascading to a nested object. For example, if the author provided information that could be used to download an external application to be used to process the object, then the user agent SHOULD attempt to download and install the application. If the user selects to not install the application, the user agent SHOULD continue to process the nested object or objects, if they exist.

The following example shows a minimally coded object [p.101] element. The @src [p.86] attribute specifies the location of the object data:

Example

```
<object src="http://www.example.com/foo.mp3">
    <em>alternate text</em>
</object>
```

The following example shows an object [p.101] element coded to process an image. The @src [p.86] attribute specifies the location of the object data, in this case the image to be processed, and the @srctype [p.86] attribute specifies the media type associated with the object data:

Example

```
<object src="http://www.example.com/foo.jpg" srctype="image/jpeg">
   <em>alternate text</em>
</object>
```

The following example shows how an applet element can be converted to an object [p.101] element. The codebase attribute is replaced with the xml:base attribute. The code attribute is replaced with the @src [p.86] attribute. The width and the height of the applet are defined using

CSS. The param [p.106] elements are not modified since the values within the param [p.106] elements are passed directly to the external application. If a particular version reference is required, that would be appended to the content of the type attribute. For example, type="application/x-java-applet;version=1.4.1"

If the archive attribute is used, the object handler should process the search order by interpreting the archive attribute value first and then the xml:base attribute value.

Example

```
<applet
  codebase="http://www.example.com/applets/classes"
  code="Clock.class"
  width="150"
  height="150">
    <param name="bgcolor" value="ffffff"/>
    <param name="border" value="5"/>
    <param name="ccolor" value="dddddd"/>
    <param name="cfont" value="TimesRoman|BOLD|18"/>
    <param name="delay" value="100"/>
    <param name="hhcolor" value="0000ff"/>
    <param name="link" value="http://www.example.com/"/>
    <param name="mhcolor" value="00ff00"/>
    <param name="ncolor" value="000000"/>
    <param name="nradius" value="80"/>
    <param name="shcolor" value="ff0000"/>
</applet>
```

Example

```
<style type="text/css">
#obj1 {width:150px; height:150px;}
</style>
...
<object id="obj1"
  xml:base="http://www.example.com/applets/classes"
  srctype="application/x-java-applet"
  src="Clock.class">
    <param  name="delay"   value="100"/>
    <param  name="link"    value="http://www.example.com/"/>
    <param  name="border"  value="5"/>
    <param  name="nradius" value="80"/>
    <param  name="cfont"   value="TimesRoman|BOLD|18"/>
    <param  name="bgcolor" value="ddddff"/>
    <param  name="shcolor" value="ff0000"/>
    <param  name="mhcolor" value="00ff00"/>
    <param  name="hhcolor" value="0000ff"/>
    <param  name="ccolor"  value="dddddd"/>
    <param  name="ncolor"  value="000000"/>
    <em>alternate text</em>
</object>
```

Authors should always include alternate text as the content of the object [p.101] element declaration when an embedded object is not defined.

The following example demonstrates how alternate text may be used within an object [p.101] element.

Example

```
<object src="http://www.example.com/foo.mp3" srctype="audio/mpeg">
   A really cool audio file. If you want to download and install
   a plug-in to listen to this file, please go to
   <a href="http://www.example.com">www.example.com</a>
</object>
```

In the following example, we embed several object [p.101] element declarations to illustrate how alternate processing works. In the following order: (1) an Earth applet, (2) an animation of the Earth, (3) an image of the Earth, (4) alternate text.

Example

```
<!-- First, try the applet -->
<object
    src="http://www.example.com/applets/classes/TheEarth.class"
    srctype="application/x-java-applet">
      <!-- Else, try the video -->
      <object
          src="TheEarth.mpeg"
          srctype="video/mpeg"
          xml:base="http://www.example.com/videos/">
            <!-- Else, try the image -->
            <object
                src="TheEarth.png"
                srctype="image/png"
                xml:base="http://www.example.com/images/">
                <!-- Else process the alternate text -->
                The <strong>Earth</strong> as seen from space.
            </object>
      </object>
</object>
```

The outermost object [p.101] element declaration specifies an applet that requires no initial values, the @src [p.86] attribute points to the applet class file, and the @srctype [p.86] attribute defines the media type. An @xml:base [p.68] attribute could have been used to point to the base location to access the class file. In this example, however, the @src [p.86] attribute value contains an absolute URL so the @xml:base [p.68] attribute was not required. An @archive [p.101] attribute could have been used if the author needed to include any associated files. The second object [p.101] element declaration specifies an MPEG animation, and the @xml:base [p.68] attribute defines the location of the object data defined in the @src [p.86] attribute. We also set the @srctype [p.86] attribute so that a user agent can determine if it has the capability to process the object data or to invoke an external application to process the MPEG. The third object element declaration specifies a PNG file and furnishes alternate text in case all other mechanisms fail.

Inline vs. external data. *Data to be processed may be supplied in two ways: inline and from an external resource. While the former method will generally lead to faster processing, it is not convenient when processing large quantities of data.*

25.2. The param element

Attributes

name = CDATA [p.29]
: This attribute defines the name of a run-time parameter, assumed to be known by the object handler. Whether the property name is case-sensitive depends on the specific object handler implementation.

value = CDATA [p.29]
: This attribute specifies the value of a run-time parameter specified by @name [p.106] . Property values have no meaning to XHTML; their meaning is determined by the object in question.

valuetype = `data` | `ref` | `object`
: This attribute specifies the type of the `value` attribute.

  Possible values:

  - `data`: This is the default value for the attribute. It means that the value specified by @value [p.106] will be evaluated and passed to the object's implementation as a string.
  - `ref`: The value specified by @value [p.106] is a URI that designates a resource where run-time values are stored. This allows support tools to identify URIs given as parameters. The URI must be passed to the object **as is**, i.e., unresolved.
  - `object`: The value specified by @value [p.106] is an identifier that refers to an object [p.101] declaration in the same document. The identifier must be the value of the @id [p.62] attribute set for the declared object [p.101] element.

type = ContentType [p.29]
: This attribute specifies the content type of the resource designated by the @value [p.106] attribute only in the case where @valuetype [p.106] is set to "ref". This attribute thus specifies for the user agent, the type of values that will be found at the URI designated by value.

param [p.106] elements specify a set of values that may be required to process the object data by an object handler at run-time. Any number of param [p.106] elements may appear in the content of an object [p.101] element, in any order, but must be placed at the start of the content of the enclosing object [p.101] element, with the exception of optional caption [p.81] and standby [p.110] elements.

The syntax of names and values is assumed to be understood by the user agent or the external application that will process the object data. This document does not specify how object handlers should retrieve name/value pairs nor how they should interpret parameter names that

appear twice.

The user agent or the external application can utilize the param [p.106] element name/value pairs to pass unique datapoints to trigger specific functions or actions. For example, the user agent may wish to trigger an external application download if the user does not have an appropriate application installed on their system.

We return to the clock example to illustrate the use of the param [p.106] element. For example, suppose that the applet is able to handle two run-time parameters that define its initial height and width. We can set the initial dimensions to 40x40 pixels with two param [p.106] elements.

Example

```
<object
    src="http://www.example.com/myclock.class"
    srctype="application/x-java-applet">
        <param name="height" value="40" valuetype="data" />
        <param name="width" value="40" valuetype="data" />
        This user agent cannot process a java applet.
</object>
```

In the following example, run-time data for the object's "Init_values" parameter is specified as an external resource (a GIF file). The value of the @valuetype [p.106] attribute is thus set to "ref" and the @value [p.106] is a URI designating the resource.

Example

```
<object
    src="http://www.example.com/gifappli"
    srctype="image/gif">
        <standby>Loading Elvis...</standby>
        <param name="Init_values"
            value="./images/elvis.gif"
            valuetype="ref" />
        Elvis lives!
</object>
```

Note that we have also set the standby [p.110] element so that the object handler may display a message while the object data is downloading.

When an object [p.101] element is processed, the user agent must search the content for only those param [p.106] elements that are direct children and "feed" them to the object handler.

Thus, in the following example, if "obj1" is processed, then the name/value content of "param1" applies to "obj1" (and not "obj2"). If "obj1" is not processed and "obj2" is, "param1" is ignored, and the name/value content of "param2" applies to "obj2". If neither object [p.101] element is processed, neither param [p.106] name/value content applies.

Example

```
<object
    src="obj1"
    srctype="application/x-something">
    <param name="param1" value="value1" />
    <object
        src="obj2"
        srctype="application/x-something">
        <param name="param2" value="value2" />
        This user agent cannot process this application.
    </object>
</object>
```

25.2.1. Referencing object data

The location of an object's data is given by a URI. The URI may be either an absolute URI or a relative URI. If the URI is relative, it may be based from the referring document location or from the @xml:base [p.68] attribute location.

In the following example, we insert a video clip into an XHTML document.

Example

```
<object
    src="mymovie.mpg"
    srctype="video/mpeg">
    A film showing how to open the printer to replace the cartridge.
</object>
```

By setting the @srctype [p.86] attribute, a user agent can determine whether to retrieve the external application based on its ability to do so. The location of the object data is relative to the referencing document, in this example the object data would need to exist within the same directory.

The following example specifies a base location via the @xml:base [p.68] attribute. The @src [p.86] attribute defines the data to process.

Example

```
<object
    xml:base="http://www.example.com/"
    src="mymovie.mpg"
    srctype="video/mpeg">
    This user agent cannot process this movie.
</object>
```

25.2.2. Object element declarations and instantiations

The following example is for illustrative purposes only. When a document is designed to contain more than one instance of the same object data, it is possible to separate the declaration of the object from the references to the object data. Doing so has several advantages:

- The object data may be retrieved from the network by the object handler *one time* (during the declaration) and reused for each additional reference to that object data.
- It is possible to reference the object data from a location other than the object element in which it was defined, for example, from a link.
- It is possible to specify an object data as run-time data for other object element declarations.

To declare an object element so that it is not executed when read by the object handler, set the boolean @declare [p.101] attribute in the object [p.101] element. At the same time, authors must identify the object declaration by setting the @id [p.62] attribute in the object [p.101] element to a unique value. Later processing of the object data will refer to this identifier.

A declared object [p.101] element must appear in a document before the first time the object data is referenced. For example, the declaring object element must appear before a link referencing the object data.

When an object element is defined with the @declare [p.101] attribute, the object handler is instantiated every time an element refers to that object data later in the document. The references will require the object data to be processed (e.g., a link that refers to it is activated, an object element that refers to it is activated, etc.).

In the following example, we declare an object [p.101] element and cause the object handler to be instantiated by referring to it from a link. Thus, the object data can be activated by clicking on some highlighted text, for example.

Example

```
<object
    declare="declare"
    id="earth.declaration"
    src="TheEarth.mpg"
    srctype="video/mpeg">
    The <strong>Earth</strong> as seen from space.
</object>
<em>...later in the document...</em>
<p>A neat <a href="#earth.declaration">animation of The Earth!</a></p>
```

In the previous example, when the document is initially loaded the object data should not be processed. If this was to be processed within a visual user agent, the object data would not be displayed. When the user selects the anchor data, the object data would then be initialized and displayed. This would also be the case for an audio file, where the file would be instantiated but would not be processed. Selecting the anchor data would then trigger the audio file to be processed.

User agents that do not support the @declare [p.101] attribute must process the contents of the object [p.101] element.

25.3. The standby element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

The standby [p.110] element specifies a message that a user agent may render while loading the object's implementation and data.

26. XHTML Role Attribute Module

This section is *normative* for purposes of defining the integration of the XHTML Role Module into XHTML 2. The semantics of the XHTML Role Module itself are normatively defined in [XHTMLROLE [p.311]].

This module defines the role attribute and the associated Role Attribute Collection. The Role Attribute Collection is included in the Common [p.34] attribute collection.

26.1. Role Attribute Collection

Attribute	Notes
role (CURIE [p.30] s)	

The XHTML Role Attribute allows the author to annotate XHTML Documents with machine-extractable semantic information about the purpose of an element. Use cases include accessibility, device adaptation, server-side processing, and complex data description.

Implementations: RelaxNG [p.203] XML Schema [p.283]

27. Ruby Module

This section is *normative* for purposes of defining the integration of the Ruby Module into XHTML 2. The semantics of the Ruby Module itself are normatively defined in [RUBY [p.310]].

This module adds the ruby element to the Text [p.47] content set of the Text [p.47] Module. XHTML 2.0 supports the maximal content model for the `ruby` element, defined as follows:

```
((rb, (rt | (rp, rt, rp))) | (rbc, rtc, rtc?))
```

As defined in [RUBY [p.310]], the `ruby` element is not allowed to nest.

Implementations: RELAX NG [p.208] , XML Schema [p.283] , DTD

28. XHTML Style Sheet Module

This section is *normative*.

The Style Sheet Module defines an element to be used when declaring internal style sheets. The element and attributes defined by this module are:

Elements	Attributes	Content Model
style [p.115]	Common [p.34] , @disabled [p.115] ("disabled"), @media [p.115] (MediaDesc [p.31])	PCDATA

Implementations: RELAX NG [p.190] , XML Schema [p.276]

28.1. The style element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

disabled = "`disabled`"
: When present, this boolean attribute makes the current element inactive (e.g., a "disabled" style element would have its styles removed from the active style collection).

media = MediaDesc [p.31]
: The value of this attribute is a comma-separated list of media descriptors for which this `access` element is intended. When the value of this attribute matches the current processing media, the associated `access` element is considered *active* and processed normally; otherwise it is *inactive* and ignored. The default value for this attribute is `all`.

The style [p.115] element allows an author to put style sheet rules in the head of the document. XHTML permits any number of style [p.115] elements in the head [p.36] section of a document.

The syntax of style data depends on the style sheet language.

Rules for style rule precedences and inheritance depend on the style sheet language.

Example

```
<style type="text/css">
  h1 {border-width: thin; border-style: solid; text-align: center}
</style>
```

28.1.1. External style sheets

Authors may separate style sheets from XHTML documents. This offers several benefits:

- Authors and web site managers may share style sheets across a number of documents (and sites).
- Authors may change the style sheet without requiring modifications to the document.
- User agents may load style sheets selectively (based on media descriptors).

28.1.2. Preferred and alternate style sheets

XHTML allows authors to associate any number of external style sheets with a document. The style sheet language defines how multiple external style sheets interact (for example, the CSS "cascade" rules).

Authors may specify a number of mutually exclusive style sheets called *alternate* style sheets. Users may select their favorite among these depending on their preferences. For instance, an author may specify one style sheet designed for small screens and another for users with weak vision (e.g., large fonts). User agents should allow users to select from alternate style sheets.

The author may specify that one of the alternates is a *preferred* style sheet. User agents should apply the author's preferred style sheet unless the user has selected a different alternate.

Authors may group several alternate style sheets (including the author's preferred style sheets) under a single *style name*. When a user selects a named style, the user agent must apply all style sheets with that name. User agents must not apply alternate style sheets with a different style name. The section on specifying external style sheets explains how to name a group of style sheets.

Authors may also specify *persistent* style sheets that user agents must apply in addition to any alternate style sheet.

User agents must respect media descriptors [p.31] when applying any style sheet.

User agents should also allow users to disable the author's style sheets entirely, in which case the user agent must not apply any persistent or alternate style sheets.

28.1.3. Specifying external style sheets

Authors specify external style sheets using the `xml-stylesheet` processing instruction [XMLSTYLE [p.312]], or, for CSS, by using the `@import` facility within a style [p.115] element.

User agents should provide a means for users to view and pick from the list of alternate styles, if specified.

In this example, we first specify a persistent style sheet located in the file `mystyle.css`:

Example

```
<?xml-stylesheet href="mystyle.css" type="text/css"?>
```

Setting the `title` pseudo-attribute makes this the author's preferred style sheet:

Example

```
 <?xml-stylesheet href="mystyle.css" title="compact" type="text/css"?>
```

Adding the `alternate` pseudo-attribute makes it an alternate style sheet:

Example

```
<?xml-stylesheet href="mystyle.css" title="Medium" alternate="yes" type="text/css"?>
```

29. XHTML Style Attribute Module

This section is *normative*.

The Style Attribute Module defines the `style` attribute.

Note: use of the @style [p.119] attribute is strongly discouraged in favor of the style [p.115] element and external style sheets. In addition, content developers are advised to avoid use of the @style [p.119] attribute on content intended for use on small devices, since those devices may not support the use of in-line styles.

29.1. Style Attribute Collection

style = CDATA [p.29]

This attribute specifies style information for the current element.

The syntax of the value of the @style [p.119] attribute is determined by the default style sheet language.

This CSS example sets color and font size information for the text in a specific paragraph.

Example

```
<p style="font-size: 12pt; color: fuchsia">
   Aren't style sheets wonderful?</p>
```

In CSS, property declarations have the form "name : value" and are separated by a semi-colon.

To specify style information for more than one element, authors should use the style [p.115] element. For optimal flexibility, authors should define styles in external style sheets.

Implementations: RELAX NG [p.190] , XML Schema [p.277]

30. XHTML Tables Module

This section is *normative*.

The Tables Module provides elements for marking up tabular information in a document.

The module supports the following elements, attributes, and content model:

Elements	Attributes	Content Model
table	Common [p.34]	caption [p.81] ?, title [p.37] ?, summary [p.124] ?, (col [p.122] * | colgroup [p.122] *), ((thead [p.140] ?, tfoot [p.140] ?, tbody [p.135] +) | (tr [p.141] +))
summary	Common [p.34]	(Flow [p.39])*
col	Common [p.34] , @span [p.122] (Number [p.31])	EMPTY
colgroup	Common [p.34] , @span [p.122] (Number [p.31])	col [p.122] *
thead	Common [p.34]	tr [p.141] +
tfoot	Common [p.34]	tr [p.141] +
tbody	Common [p.34]	tr [p.141] +
tr	Common [p.34] , Forms [p.144]	(td [p.135] | th [p.135])+
td	Common [p.34] , @abbr [p.135] (Text [p.31]), @axis [p.135] (CDATA [p.29]), @colspan [p.135] (Number [p.31]), @headers [p.135] (IDREFS [p.29]), @rowspan [p.136] (Number [p.31]), @scope [p.136] ("row", "col", "rowgroup", "colgroup")	(Flow [p.39])*
th	Common [p.34] , @abbr [p.135] (Text [p.31]), @axis [p.135] (CDATA [p.29]), @colspan [p.135] (Number [p.31]), @headers [p.135] (IDREFS [p.29]), @rowspan [p.136] (Number [p.31]), @scope [p.136] ("row", "col", "rowgroup", "colgroup")	(Flow [p.39])*

Implementations: RELAX NG [p.191] , XML Schema [p.277]

30.1. The col and colgroup elements

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

span = Number [p.31]
: This attribute must be an integer > 0; the default value is 1. This specifies the number of columns in a colgroup [p.122] , or specifies the number of columns "spanned" by the col [p.122] element.

  Values mean the following:

  - In the absence of a @span [p.122] attribute, each colgroup [p.122] defines a column group containing one column.
  - If the @span [p.122] attribute is used with the colgroup [p.122] element and the value is set to N > 0, that defines a column group containing N columns.
  - If the @span [p.122] attribute is used with the col [p.122] element and the value is set to N > 1, the current col [p.122] element shares its attributes with the next N-1 columns.

  User agents must ignore this attribute if the colgroup [p.122] element contains one or more col [p.122] elements. Instead, the value must be computed by summing the span attributes of the enclosed col [p.122] elements.

The colgroup [p.122] element allows authors to create structural divisions within a table. Authors may highlight this structure through style sheets. For example, the author may wish to divide the columns into logical groups such as the student’s permanent address, phone number and emergency contact information. And group the student’s local address, phone and email address into another logical group.

A table [p.125] may either contain a single implicit column group (no colgroup [p.122] element delimits the columns) or any number of explicit column groups (each delimited by an instance of the colgroup [p.122] element).

The col [p.122] element allows authors to share attributes among several columns without implying any structural grouping. The "span" of the col [p.122] element is the number of columns that will share the element’s attributes. For example, the author may wish to apply a specific style to the student’s permanent data and apply a different style to the student’s local data.

The colgroup [p.122] element creates an explicit column group. The number of columns in the column group may be specified in two, mutually exclusive ways:

1. The colgroup @span [p.122] attribute (default value 1) specifies the number of columns in the group.
2. Each embedded col [p.122] element in the colgroup [p.122] represents one or more columns in the group.

The advantage of using the colgroup [p.122] element is that authors may logically group multiple columns. By grouping columns, the author can apply rules across the entire group. The author can also apply column width balancing across the group of columns. For example, if the author has a table with five columns and the author divides the table into two column groups, one with three columns and the other with two columns. The author could define the first column group to consume 300 pixels and the second column group to consume 100 pixels. Each column within the first column group would be 100 pixels wide and the remaining two columns would be 50 pixels wide. If the author added embedded col [p.122] elements, she could force one or more columns to be a specific width and the remaining columns within the group would be evenly divided within the remaining allotted width.

For example, the following table defines a column group and embedded columns with differing widths.

Example

```
<style type="text/css">
#colgrp1 { width: 300px }
#col1 { width: 100px }
#col2 { width: 50px }
</style>
...
<table>
  <colgroup id="colgrp1">
    <col id="col1" span="3"/>
    <col id="col2" span="2"/>
  </colgroup>
<em>...the rest of the table...</em>
</table>
```

In this example, the defined width for the colgroup [p.122] constrains all of the columns to fit within that value regardless of the of the defined values within the col [p.122] elements. In this example, the width of the columns within the column group must be constrained to fit the defined width of the column group.

When it is necessary to single out a column (e.g., for style information, to specify width information, etc.) within a group, authors must identify that column with a col [p.122] element.

The col [p.122] element allows authors to group together attribute specifications for table columns. The col [p.122] does **not** group columns together structurally -- that is the role of the colgroup [p.122] element. col [p.122] elements are empty and serve only as a support for attributes. They may appear inside or outside an explicit column group (i.e., colgroup [p.122] element).

30.1.1. Calculating the number of columns in a table

There are two ways to determine the number of columns in a table (in order of precedence):

1. If the table [p.125] element contains any colgroup [p.122] or col [p.122] elements, user agents should calculate the number of columns by summing the following:
   - For each col [p.122] element, take the value of its @span [p.122] attribute (default value 1).
   - For each colgroup [p.122] element containing at least one col [p.122] element, ignore the @span [p.122] attribute for the colgroup [p.122] element. For each col [p.122] element, perform the calculation of step 1.
   - For each empty colgroup [p.122] element, take the value of its @span [p.122] attribute (default value 1).
2. Otherwise, if the table [p.125] element contains no colgroup [p.122] or col [p.122] elements, user agents should base the number of columns on what is required by the rows. The number of columns is equal to the number of columns required by the row with the most columns, including cells that span multiple columns. For any row that has fewer than this number of columns, the end of that row should be padded with empty cells. The "end" of a row depends on the directionality of a table.

It is an error if a table contains colgroup [p.122] or col [p.122] elements and the two calculations do not result in the same number of columns.

Once the user agent has calculated the number of columns in the table, it may group them into a colgroup [p.122] .

30.2. The summary element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

This element provides a summary of the table’s purpose and structure for user agents rendering to non-visual media such as speech and Braille.

User agents MUST provide access to the content of the summary [p.124] element. As an example, access could be provided through a menu option, a mouse-over function, or through a dialog.

The following example demonstrates the difference between a table caption and a table summary.

Example

```
<table>
  <caption>Student Class Roster</caption>
  <summary>The table defines the class roster.
    The columns contain the following data:
    students name, permanent address, permanent phone,
    local address, local phone,
    declared major, assigned academic advisor,
    student standing</summary>
  <em>...the rest of the table...</em>
</table>
```

30.3. The table element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

The table [p.125] element contains all other elements that specify the caption, column groups, columns, rows, and content for a table.

30.3.1. Visual Rendering

All style associated with table rendering MUST use proper CSS2 properties.

Although CSS2 is not required, the equivalent effect MUST BE followed and integrated into the rendering model.

The following informative list describes what operations visual user agents may carry out when rendering a table:

- Provide access to the content of the summary [p.124] element. As an example, access could be provided through a menu option, a mouse-over function, or through a dialog. Authors should provide a summary of a table's content and structure so that people using non-visual user agents may better understand it.
- Render the caption, if one is defined. The caption may be rendered, for example, either on the top or the bottom of the table.
- Render the table header, if one is specified. Render the table footer, if one is specified. User agents must know where to render the header and footer. For instance, if the output medium is paged, user agents may put the header at the top of each page and the footer at the bottom. Similarly, if the user agent provides a mechanism to scroll the rows, the header may appear at the top of the scrolled area and the footer at the bottom.
- Calculate the number of columns [p.124] in the table. Note that the number of rows in a table is equal to the number of tr [p.141] elements contained by the table [p.125] element.

- Group the columns according to any column groups [p.122] specifications.
- Render the cells, row by row and grouped in appropriate columns, between the header and footer. Visual user agents should format the table according to XHTML attributes and style sheet specification.

30.3.2. Table directionality

The directionality of a table is either the inherited directionality (the default is left-to-right) or that specified by the @dir [p.79] attribute for the table [p.125] element.

For a left-to-right table, column zero is on the left side and row zero is at the top. For a right-to-left table, column zero is on the right side and row zero is at the top.

When a user agent allots extra cells to a row, extra row cells are added to the right of the table for left-to-right tables and to the left side for right-to-left tables.

Note that table [p.125] is the only element on which @dir [p.79] reverses the visual order of the columns; a single table row (tr [p.141]) or a group of columns (colgroup [p.122]) cannot be independently reversed.

When set for or inherited by the table [p.125] element, the @dir [p.79] attribute also affects the direction of text within table cells (since the @dir [p.79] attribute is inherited by block-level elements).

To specify a right-to-left table, set the @dir [p.79] attribute as follows:

Example

```
<table dir="rtl">
<em>...the rest of the table...</em>
</table>
```

The direction of text in individual cells can be changed by setting the @dir [p.79] attribute in an element that defines the cell.

30.3.3. Table rendering by non-visual user agents

This section provides more detailed discussion on cell header data and how non-visual agents may utilize that information.

30.3.3.1. Associating header information with data cells

Non-visual user agents such as speech synthesizers and Braille-based devices may use the following td [p.135] and th [p.135] element attributes to render table cells more intuitively:

- For a given data cell, the @headers [p.135] attribute lists which cells provide pertinent header information. For this purpose, each header cell must be named using the @id [p.62] attribute. Note that it's not always possible to make a clean division of cells into headers or

data. You should use the td [p.135] element for such cells together with the @id [p.62] or @scope [p.136] attributes as appropriate.

- For a given header cell, the @scope [p.136] attribute tells the user agent the data cells for which this header provides information. Authors may choose to use this attribute instead of @headers [p.135] according to which is more convenient; the two attributes fulfill the same function. The @headers [p.135] attribute is generally needed when headers are placed in irregular positions with respect to the data they apply to.
- The @abbr [p.135] attribute specifies an abbreviated header for header cells so that user agents may render header information more rapidly.

In the following example, we assign header information to cells by setting the @headers [p.135] attribute. Each cell in the same column refers to the same header cell (via the @id [p.62] attribute).

Example

```
<table>
<caption>Cups of coffee consumed by each senator</caption>
<summary>This table charts the number of cups
          of coffee consumed by each senator, the type
          of coffee (decaf or regular), and whether
          taken with sugar.</summary>
<tbody>
   <tr>
      <th id="t1">Name</th>
      <th id="t2">Cups</th>
      <th id="t3" abbr="Type">Type of Coffee</th>
      <th id="t4">Sugar?</th>
   </tr>
   <tr>
      <td headers="t1">T. Sexton</td>
      <td headers="t2">10</td>
      <td headers="t3">Espresso</td>
      <td headers="t4">No</td>
   </tr>
   <tr>
      <td headers="t1">J. Dinnen</td>
      <td headers="t2">5</td>
      <td headers="t3">Decaf</td>
      <td headers="t4">Yes</td>
   </tr>
</tbody>
</table>
```

A speech synthesizer might render this table as follows:

Example

```
Caption: Cups of coffee consumed by each senator
Summary: This table charts the number of cups
         of coffee consumed by each senator, the type
         of coffee (decaf or regular), and whether
         taken with sugar.
Name: T. Sexton,   Cups: 10,   Type: Espresso,   Sugar: No
Name: J. Dinnen,   Cups: 5,    Type: Decaf,      Sugar: Yes
```

Note how the header "Type of Coffee" is abbreviated to "Type" using the @abbr [p.135] attribute.

Here is the same example substituting the @scope [p.136] attribute for the @headers [p.135] attribute. Note the value "col" for the @scope [p.136] attribute, meaning "all cells in the current column":

Example

```
<table>
<caption>Cups of coffee consumed by each senator</caption>
<summary>
  This table charts the number of cups
  of coffee consumed by each senator, the type
  of coffee (decaf or regular), and whether
  taken with sugar.
</summary>
<tbody>
   <tr>
      <th scope="col">Name</th>
      <th scope="col">Cups</th>
      <th scope="col" abbr="Type">Type of Coffee</th>
      <th scope="col">Sugar?</th>
   </tr>
   <tr>
      <td>T. Sexton</td>
      <td>10</td>
      <td>Espresso</td>
      <td>No</td>
   </tr>
   <tr>
      <td>J. Dinnen</td>
      <td>5</td>
      <td>Decaf</td>
      <td>Yes</td>
   </tr>
</tbody>
</table>
```

Here’s a somewhat more complex example illustrating other values for the @scope [p.136] attribute:

Example

```
<table>
<summary>
  History courses offered in the community of
  Bath arranged by course name, tutor, summary,
  code, and fee
</summary>
<thead>
  <tr>
    <th colspan="5" scope="colgroup">Community Courses -- Bath Autumn 1997</th>
  </tr>
</thead>
<tbody>
  <tr>
    <th scope="col" abbr="Name">Course Name</th>
    <th scope="col" abbr="Tutor">Course Tutor</th>
    <th scope="col">Summary</th>
    <th scope="col">Code</th>
    <th scope="col">Fee</th>
  </tr>
  <tr>
    <td scope="row">After the Civil War</td>
    <td>Dr. John Wroughton</td>
    <td>
       The course will examine the turbulent years in England
       after 1646. <em>6 weekly meetings starting Monday 13th
       October.</em>
    </td>
    <td>H27</td>
    <td>&amp;amp;pound;32</td>
  </tr>
  <tr>
    <td scope="row">An Introduction to Anglo-Saxon England</td>
    <td>Mark Cottle</td>
    <td>
       One day course introducing the early medieval
       period reconstruction the Anglo-Saxons and
       their society. <em>Saturday 18th October.</em>
    </td>
    <td>H28</td>
    <td>&amp;amp;pound;18</td>
  </tr>
  <tr>
    <td scope="row">The Glory that was Greece</td>
    <td>Valerie Lorenz</td>
    <td>
     Birthplace of democracy, philosophy, heartland of theater, home of
     argument. The Romans may have done it but the Greeks did it
     first. <em>Saturday day school 25th October 1997</em>
    </td>
    <td>H30</td>
    <td>&amp;amp;pound;18</td>
  </tr>
</tbody>
</table>
```

A graphical user agent might render this as:

Community Courses -- Bath Autumn 1997				
Course Name	**Course Tutor**	**Summary**	**Code**	**Fee**
After the Civil War	Dr. John Wroughton	The course will examine the turbulent years in England after 1646. *6 weekly meetings starting Monday 13th October.*	H27	£32
An Introduction to Anglo-Saxon England	Mark Cottle	One day course introducing the early medieval period reconstruction the Anglo-Saxons and their society. *Saturday 18th October.*	H28	£18
The Glory that was Greece	Valerie Lorenz	Birthplace of democracy, philosophy, heartland of theater, home of argument. The Romans may have done it but the Greeks did it first. *Saturday day school 25th October 1997*	H30	£18

Note the use of the @scope [p.136] attribute with the "row" value. Although the first cell in each row contains data, not header information, the @scope [p.136] attribute makes the data cell behave like a row header cell. This allows speech synthesizers to provide the relevant course name upon request or to state it immediately before each cell's content.

30.3.3.2. Categorizing cells

Users browsing a table with a speech-based user agent may wish to hear an explanation of a cell's contents in addition to the contents themselves. One way the user might provide an explanation is by speaking associated header information before speaking the data cell's contents (see the section on associating header information with data cells [p.126]).

Users may also want information about more than one cell, in which case header information provided at the cell level (by @headers [p.135] , @scope [p.136] , and @abbr [p.135]) may not provide adequate context. Consider the following table, which classifies expenses for meals, hotels, and transport in two locations (San Jose and Seattle) over several days:

Travel Expense Report

	Meals	Hotels	Transport	subtotals
San Jose				
25-Aug-97	37.74	112.00	45.00	
26-Aug-97	27.28	112.00	45.00	
subtotals	65.02	224.00	90.00	379.02
Seattle				
27-Aug-97	96.25	109.00	36.00	
28-Aug-97	35.00	109.00	36.00	
subtotals	131.25	218.00	72.00	421.25
Totals	196.27	442.00	162.00	**800.27**

Users might want to extract information from the table in the form of queries:

- "What did I spend for all my meals?"
- "What did I spend for meals on 25 August?"
- "What did I spend for all expenses in San Jose?"

Each query involves a computation by the user agent that may involve zero or more cells. In order to determine, for example, the costs of meals on 25 August, the user agent must know which table cells refer to "Meals" (all of them) and which refer to "Dates" (specifically, 25 August), and find the intersection of the two sets.

To accommodate this type of query, the table model allows authors to place cell headers and data into categories. For example, for the travel expense table, an author could group the header cells "San Jose" and "Seattle" into the category "Location", the headers "Meals", "Hotels", and "Transport" in the category "Expenses", and the four days into the category "Date". The previous three questions would then have the following meanings:

- "What did I spend for all my meals?" means "What are all the data cells in the "Expenses=Meals" category?
- "What did I spend for meals on 25 August?" means "What are all the data cells in the "Expenses=Meals" and "Date=Aug-25-1997" categories?
- "What did I spend for all expenses in San Jose?" means "What are all the data cells in the "Expenses=Meals, Hotels, Transport" and "Location=San Jose" categories?

Authors categorize a header or data cell by setting the @axis [p.135] attribute for the cell. For instance, in the travel expense table, the cell containing the information "San Jose" could be placed in the "Location" category as follows:

Example

```
<th id="a6" axis="location">San Jose</th>
```

Any cell containing information related to "San Jose" should refer to this header cell via either the @headers [p.135] or the @scope [p.136] attribute. Thus, meal expenses for 25-Aug-1997 should be marked up to refer to @id [p.62] attribute (whose value here is "a6") of the "San Jose" header cell:

Example

```
<td headers="a6">37.74</td>
```

Each @headers [p.135] attribute provides a list of @id [p.62] references. Authors may thus categorize a given cell in any number of ways (or, along any number of "headers", hence the name).

Below we mark up the travel expense table with category information:

Example

```
<table>
<caption>Travel Expense Report</caption>
<summary>
  This table summarizes travel expenses
  incurred during August trips to
  San Jose and Seattle
</summary>
<tbody>
   <tr>
      <th></th>
      <th id="a2" axis="expenses">Meals</th>
      <th id="a3" axis="expenses">Hotels</th>
      <th id="a4" axis="expenses">Transport</th>
      <td>subtotals</td>
   </tr>
   <tr>
      <th id="a6" axis="location">San Jose</th>
      <th></th>
      <th></th>
      <th></th>
      <td></td>
   </tr>
   <tr>
      <td id="a7" axis="date">25-Aug-97</td>
      <td headers="a6 a7 a2">37.74</td>
      <td headers="a6 a7 a3">112.00</td>
      <td headers="a6 a7 a4">45.00</td>
      <td></td>
   </tr>
   <tr>
      <td id="a8" axis="date">26-Aug-97</td>
      <td headers="a6 a8 a2">27.28</td>
      <td headers="a6 a8 a3">112.00</td>
      <td headers="a6 a8 a4">45.00</td>
      <td></td>
   </tr>
   <tr>
      <td>subtotals</td>
```

```
      <td>65.02</td>
      <td>224.00</td>
      <td>90.00</td>
      <td>379.02</td>
   </tr>
   <tr>
      <th id="a10" axis="location">Seattle</th>
      <th></th>
      <th></th>
      <th></th>
      <td></td>
   </tr>
   <tr>
      <td id="a11" axis="date">27-Aug-97</td>
      <td headers="a10 a11 a2">96.25</td>
      <td headers="a10 a11 a3">109.00</td>
      <td headers="a10 a11 a4">36.00</td>
      <td></td>
   </tr>
   <tr>
      <td id="a12" axis="date">28-Aug-97</td>
      <td headers="a10 a12 a2">35.00</td>
      <td headers="a10 a12 a3">109.00</td>
      <td headers="a10 a12 a4">36.00</td>
      <td></td>
   </tr>
   <tr>
      <td>subtotals</td>
      <td>131.25</td>
      <td>218.00</td>
      <td>72.00</td>
      <td>421.25</td>
   </tr>
   <tr>
      <th>Totals</th>
      <td>196.27</td>
      <td>442.00</td>
      <td>162.00</td>
      <td>800.27</td>
   </tr>
</tbody>
</table>
```

Note that marking up the table this way also allows user agents to avoid confusing the user with unwanted information. For instance, if a speech synthesizer were to speak all of the figures in the "Meals" column of this table in response to the query "What were all my meal expenses?", a user would not be able to distinguish a day’s expenses from subtotals or totals. By carefully categorizing cell data, authors allow user agents to make important semantic distinctions when rendering.

Of course, there is no limit to how authors may categorize information in a table. In the travel expense table, for example, we could add the additional categories "subtotals" and "totals".

This specification does not require user agents to handle information provided by the @axis [p.135] attribute, nor does it make any recommendations about how user agents may present @axis [p.135] information to users or how users may query the user agent about this information.

However, user agents, particularly speech synthesizers, may want to factor out information common to several cells that are the result of a query. For instance, if the user asks "What did I spend for meals in San Jose?", the user agent would first determine the cells in question (25-Aug-1997: 37.74, 26-Aug-1997:27.28), then render this information. A user agent speaking this information might read it:

Example

```
Location: San Jose. Date: 25-Aug-1997. Expenses, Meals: 37.74
Location: San Jose. Date: 26-Aug-1997. Expenses, Meals: 27.28
```

or, more compactly:

Example

```
San Jose, 25-Aug-1997, Meals: 37.74
San Jose, 26-Aug-1997, Meals: 27.28
```

An even more economical rendering would factor the common information and reorder it:

Example

```
San Jose, Meals, 25-Aug-1997: 37.74
                 26-Aug-1997: 27.28
```

User agents that support this type of rendering should allow authors a means to customize rendering (e.g., through style sheets).

30.3.3.3. Algorithm to find heading information

In the absence of header information from either the @scope [p.136] or @headers [p.135] attribute, user agents may construct header information according to the following algorithm. The goal of the algorithm is to find an ordered list of headers. (In the following description of the algorithm the table directionality [p.126] is assumed to be left-to-right.)

- First, search left from the cell's position to find row header cells. Then search upwards to find column header cells. The search in a given direction stops when the edge of the table is reached or when a data cell is found after a header cell.
- Row headers are inserted into the list in the order they appear in the table. For left-to-right tables, headers are inserted from left to right.
- Column headers are inserted after row headers, in the order they appear in the table, from top to bottom.
- If a header cell has the @headers [p.135] attribute set, then the headers referenced by this attribute are inserted into the list and the search stops for the current direction.

- td [p.135] cells that set the @axis [p.135] attribute are also treated as header cells.

30.4. The tbody element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

The tbody [p.135] element contains rows of table data. In tables that also contain thead [p.140] or tfoot [p.140] elements, all of these sections must contain the same number of columns.

30.5. The td and th elements

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

abbr = Text [p.31]
: This attribute should be used to provide an abbreviated form of the cell's content, and may be rendered by user agents when appropriate in place of the cell's content. Abbreviated names should be short since user agents may render them repeatedly. For instance, speech synthesizers may render the abbreviated headers relating to a particular cell before rendering that cell's content.

axis = CDATA [p.29]
: This attribute may be used to place a cell into conceptual categories that can be considered to form axes in an n-dimensional space. User agents may give users access to these categories (e.g., the user may query the user agent for all cells that belong to certain categories, the user agent may present a table in the form of a table of contents, etc.). Please consult the section on categorizing cells [p.130] for more information. The value of this attribute is a comma-separated list of category names.

colspan = Number [p.31]
: This attribute specifies the number of columns spanned by the current cell. The default value of this attribute is one ("1"). The value zero ("0") means that the cell spans all columns from the current column to the last column of the column group (colgroup [p.122]) in which the cell is defined.

headers = IDREFS [p.29]
: This attribute specifies the list of header cells that provide header information for the current data cell. The value of this attribute is a space-separated list of cell names; those cells must be named by setting their @id [p.62] attribute. Authors generally use the @headers [p.136] attribute to help non-visual user agents render header information about data cells (e.g., header information is spoken prior to the cell data), but the attribute may also be used in conjunction with style sheets. See also the @scope [p.136] attribute.

rowspan = Number [p.31]
: This attribute specifies the number of rows spanned by the current cell. The default value of this attribute is one ("1"). The value zero ("0") means that the cell spans all rows from the current row to the last row of the current table section (rowgroup) in which the cell is defined. thead [p.140] , tbody [p.135] , and tfoot [p.140] elements are rowgroups.

scope = `row`|`col`|`rowgroup`|`colgroup`
: This attribute specifies the set of data cells for which the current header cell provides header information. This attribute may be used in place of the @headers [p.136] attribute, particularly for simple tables. When specified, this attribute must have one of the following values:
  - **row:** The current cell provides header information for the rest of the row that contains it (see also the section on table directionality [p.126]).
  - **col:** The current cell provides header information for the rest of the column that contains it.
  - **rowgroup:** The header cell provides header information for the rest of the row group that contains it.
  - **colgroup:** The header cell provides header information for the rest of the column group [p.122] that contains it.

Table cells may contain two types of information: header information and data. This distinction enables user agents to render header and data cells distinctly, even in the absence of style sheets. For example, visual user agents may present header cell text with a bold font. Speech synthesizers may render header information with a distinct voice inflection.

The th [p.135] element defines a cell that contains header information. User agents have two pieces of header information available: the contents of the th [p.135] element and the value of the @abbr [p.135] attribute. User agents must render either the contents of the cell or the value of the @abbr [p.135] attribute. For visual media, the latter may be appropriate when there is insufficient space to render the full contents of the cell. For non-visual media @abbr [p.135] may be used as an abbreviation for table headers when these are rendered along with the contents of the cells to which they apply.

The @headers [p.136] and @scope [p.136] attributes also allow authors to help non-visual user agents process header information. Please consult the section on labeling cells for non-visual user agents [p.126] for information and examples.

The td [p.135] element defines a cell that contains data.

Cells may be empty (i.e., contain no data).

30.5.1. Cells that span several rows or columns

Cells may span several rows or columns. The number of rows or columns spanned by a cell is set by the @rowspan [p.136] and @colspan [p.135] attributes for the th [p.135] and td [p.135] elements.

In this table definition, we specify that the cell in row four, column two should span a total of three columns, including the current column.

```
<table>
<caption>Cups of coffee consumed by each senator</caption>
<tbody>
   <tr>
      <th>Name</th>
      <th>Cups</th>
      <th>Type of Coffee</th>
      <th>Sugar?</th>
   </tr>
   <tr>
      <td>T. Sexton</td>
      <td>10</td>
      <td>Espresso</td>
      <td>No</td>
   </tr>
   <tr>
      <td>J. Dinnen</td>
      <td>5</td>
      <td>Decaf</td>
      <td>Yes</td>
   </tr>
   <tr>
      <td>A. Soria</td>
      <td colspan="3"><em>Not available</em></td>
   </tr>
</tbody>
</table>
```

This table might be rendered on a tty device by a visual user agent as follows:

```
Cups of coffee consumed by each senator
 ---------------------------------------
 |   Name  |Cups|Type of Coffee|Sugar?|
 ---------------------------------------
 |T. Sexton|10  |Espresso      |No    |
 ---------------------------------------
 |J. Dinnen|5   |Decaf         |Yes   |
 ---------------------------------------
 |A. Soria |Not available              |
 ---------------------------------------
```

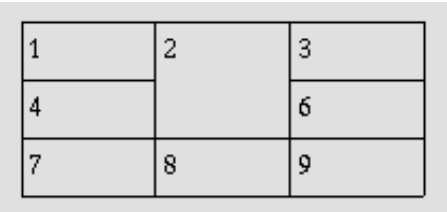
The next example illustrates (with the help of table borders) how cell definitions that span more than one row or column affect the definition of later cells. Consider the following table definition:

```
<table>
<tbody>
   <tr>
      <td>1</td>
      <td rowspan="2">2</td>
      <td>3</td>
   </tr>
   <tr>
      <td>4</td>
      <td>6</td>
   </tr>
   <tr>
      <td>7</td>
      <td>8</td>
      <td>9</td>
      <td></td>
   </tr>
</tbody>
</table>
```

As cell "2" spans the first and second rows, the definition of the second row will take it into account. Thus, the second td [p.135] in row two actually defines the row's third cell. Visually, the table might be rendered to a tty device as:

```
-------------
| 1 | 2 | 3 |
----|   |----
| 4 |   | 6 |
----|---|----
| 7 | 8 | 9 |
-------------
```

while a graphical user agent might render this as:



Note that if the td [p.135] defining cell "6" had been omitted, an extra empty cell would have been added by the user agent to complete the row.

Similarly, in the following table definition:

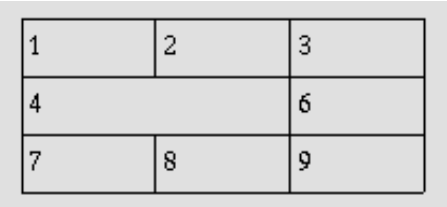
```
<table>
<tbody>
   <tr>
      <td>1</td>
```

```
      <td>2</td>
      <td>3</td>
   </tr>
   <tr>
      <td colspan="2">4</td>
      <td>6</td>
   </tr>
   <tr>
      <td>7</td>
      <td>8</td>
      <td>9</td>
   </tr>
</tbody>
</table>
```

cell "4" spans two columns, so the second td [p.135] in the row actually defines the third cell ("6"):

```
-------------
| 1 | 2 | 3 |
--------|----
| 4     | 6 |
--------|----
| 7 | 8 | 9 |
-------------
```

A graphical user agent might render this as:



Defining overlapping cells is an error. User agents may vary in how they handle this error (e.g., rendering may vary).

The following illegal example illustrates how one might create overlapping cells. In this table, cell "5" spans two rows and cell "7" spans two columns, so there is overlap in the cell between "7" and "9":

```
<table>
<tbody>
   <tr>
      <td>1</td>
      <td>2</td>
      <td>3</td>
   </tr>
   <tr>
      <td>4</td>
      <td rowspan="2">5</td>
      <td>6</td>
   </tr>
```

```
    <tr>
       <td colspan="2">7</td>
       <td>9</td>
    </tr>
</tbody>
</table>
```

30.6. The thead and tfoot elements

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

Table rows may be grouped into a table head, table foot, and one or more table body sections, using the thead [p.140] , tfoot [p.140] and tbody [p.135] elements, respectively. This division enables user agents to support scrolling of table bodies independently of the table head and foot. When long tables are printed, the table head and foot information may be repeated on each page that contains table data.

The table head and table foot should contain information about the table's columns. The table body must contain rows of table data.

When present, each thead [p.140] , tfoot [p.140] , and tbody [p.135] creates a *row group*. Each row group must contain at least one row, defined by the tr [p.141] element.

If the thead [p.140] , tfoot [p.140] , and tbody [p.135] elements are used, and a rowspan attribute is used within a group, the rowspan must remain within the group boundaries of which it is defined.

This example illustrates the order and structure of the table head, foot, and bodies.

Example

```
<table>
<thead>
     <tr> <em>...header information...</em></tr>
</thead>
<tfoot>
     <tr> <em>...footer information...</em></tr>
</tfoot>
<tbody>
     <tr> <em>...first row of block one data...</em></tr>
     <tr> <em>...second row of block one data...</em></tr>
</tbody>
<tbody>
     <tr> <em>...first row of block two data...</em></tr>
```

```
      <tr> <em>...second row of block two data...</em></tr>
      <tr> <em>...third row of block two data...</em></tr>
</tbody>
</table>
```

tfoot [p.140] must appear before tbody [p.135] within a table [p.125] definition so that user agents can render the foot before receiving all of the (potentially numerous) rows of data.

30.7. The tr element

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

The tr [p.141] elements acts as a container for a row of table cells.

This sample table contains three rows, each begun by the tr [p.141] element:

```
<table>
<caption>Cups of coffee consumed by each senator</caption>
<summary>This table charts the number of cups
  of coffee consumed by each senator, the type
  of coffee (decaf or regular), and whether
  taken with sugar.</summary>
<tbody>
   <tr> ...A header row...</tr>
   <tr> ...First row of data...</tr>
   <tr> ...Second row of data...</tr>
</tbody>
</table>
```

31. XForms Module

This section is *normative* for purposes of defining the integration of XForms into XHTML 2.0. The semantics of XForms itself is normatively defined in [XFORMS [p.311]].

The XForms Module provides a rich collection of forms features within the XHTML namespace.

The content model for XForms defines several content sets:

XForms Core
    model
XForms Form Controls
    input | secret | textarea | output | upload | range | trigger | submit | select | select1
XForms Actions
    action | dispatch | rebuild | recalculate | revalidate | refresh | setfocus | load | setvalue | send | reset | message | insert | delete | setindex
XForms Group
    group
XForms Switch
    switch
XForms Repeat
    repeat

31.1. Core XForms

The XForms Core content set is added to the content model of the head [p.36] element in the Document [p.35] module, to the Structural [p.39] content set of the Structural Module., and to the Text [p.47] content set of the Text module.

31.2. XForms Actions

The XForms Actions content set is added to the content model of the head [p.36] element in the Document [p.35] module, to the Structural [p.39] content set, and to the Text [p.47] content set.

31.3. Form Controls

The XForms Form Controls content set is added to the Structural [p.39] content set, and to the Text [p.47] content set.

The Text [p.47] content set is added to the XForms UI Inline content set, allowing various XHTML elements inside form control labels.

31.4. Group

The XForms Group content set is added to the Structural [p.39] content set, and to the Text [p.47] content set.

group [p.143] elements can freely nest.

The Structural [p.39] content set is added to the XForms Group content set.

31.5. Switch

The XForms Switch content set is added to the Structural [p.39] content set, and to the Text [p.47] content set.

Alternating switch [p.143] and case elements can freely nest.

The Structural [p.39] content set is added to the content model of case, after an optional label element.

31.6. Repeat

The XForms Repeat content set is added to the Structural [p.39] content set, and to the Text [p.47] content set.

repeat [p.143] elements can freely nest.

The Structural [p.39] content set is added to the content model of case, after an optional label element.

31.7. XForms Repeat Attribute Collection

This module also includes the XForms Repeat Attribute Collection via attributes from [XFORMS [p.311]]. The normative definition of those attributes and their semantics is included in that specification.

Attributes	Notes
repeat-model (IDREF [p.29])	
repeat-bind (IDREF [p.29])	
repeat-nodeset (LocationPath [p.31])	
repeat-startindex (Number [p.31])	
repeat-numbef (Number [p.31])	

31.8. Other Attribute Collections

XHTML 2 adds the Common [p.34] attribute collection to the XForms Common Attribute Group.

The XForms Attribute Groups for single-node binding and Nodeset binding are allowed only on the XForms elements they are defined for.

32. XML Events Module

This section is *normative* for purposes of defining the integration of XML Events into XHTML 2. The semantics of XML Events itself is normatively defined in [XMLEVENTS [p.311]].

This module includes the listener element as defined in [XMLEVENTS [p.311]]. As permitted by XML Events 2, this element is incorporated into the XHTML namespace.

32.1. Events

This module defines the Events Attribute Collection via the global attributes from [XMLEVENTS [p.311]]. The normative definition of those attributes and their semantics is included in that specification. Their names and datatypes are listed below:

Attributes	Notes
event * (QNames [p.31])	
observer (IDREFS [p.29])	
eventTarget (IDREFS [p.29])	
handler (URI [p.31])	
phase (`"bubble"` | `"capture"` | `"default"*` | `"target"`)	
propagate ("stop" | "continue"*)	
defaultAction (`cancel`|`perform*`)	

RelaxNG [p.208] , XML Schema [p.286]

33. XML Handlers Module

This section is *normative* for purposes of defining the integration of XML Handlers into XHTML 2. The semantics of XML Handlers itself is normatively defined in [XMLEVENTS [p.311]].

The XML Handlers Module defines elements that are used to contain information pertaining to event handler implementations, usually defined in a scripting language. This section defines the content model of the XML Handlers module in XHTML 2. As permitted by [XMLEVENTS [p.311]], these elements are incorporated into the XHTML namespace.

Elements and attributes included in this module are:

Element	Attributes	Content Model
action	Common [p.34] , event* (QNames [p.31]), eventTarget (IDREFS [p.29]), declare ("declare"), if (`ConditionalExpression`), while (`ConditionalExpression`)	(action | dispatchEvent | addEventListener | removeEventListener | script [p.151] | stopPropagation | preventDefault)+
dispatchEvent	Common [p.34] , eventType* (QName [p.31]), targetid (IDREFS [p.29]), bubbles ("bubbles"), cancelable ("cancelable")	EMPTY
addEventListener	Common [p.34] , event* (QName [p.31]), handler* (IDREF [p.29]), phase ("bubble" | "capture" | "default"* | "target")	EMPTY
removeEventListener	Common [p.34] , event* (QName [p.31]), handler* (IDREF [p.29]), phase ("bubble" | "capture" | "default"* | "target")	EMPTY
stopPropagation	Common [p.34] , event* (QName [p.31])	EMPTY
preventDefault	Common [p.34] , event* (QName [p.31])	EMPTY

Implementations: RelaxNG [p.210] , XML Schema [p.287]

34. XML Scripting Module

This section is *normative* for purposes of defining the integration of the XML Script Module into XHTML 2. The semantics of XML Scripting Module itself are normatively defined in [XMLEVENTS [p.311]]. As permitted by [XMLEVENTS [p.311]], this element is incorporated into the XHTML namespace.

This module defines the script element as follows:

Element	Attributes	Content Model
script	Common [p.34] , charset (Charset [p.29]), defer ("defer"), implements (URIorSafeCURIEs [p.31])	PCDATA

Implementations: RelaxNG [p.212] , XML Schema [p.289]

35. XHTML Legacy Edit Module

This section is *normative*.

This module defines the traditional HTML elements for marking up inserted and deleted content.

Element	Attributes	Content Model
del	Common [p.34]	(Text [p.47])*
ins	Common [p.34]	(Text [p.47])*

Implementations: RELAX NG [p.203] , XML Schema [p.280]

35.1. The del element

The del [p.153] element is used to indicate that a section of a document has been deleted with respect to a different version of the document (e.g., in draft legislation where lawmakers need to view the changes).

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

This example could be from a bill to change the legislation for how many deputies a County Sheriff can employ from 3 to 5.

Example

```
<p>
  A Sheriff can employ <del>3</del><ins>5</ins> deputies.
</p>
```

User agents should render deleted text in ways that make the change obvious. For instance, inserted text may appear in a special font, deleted text may not be shown at all or be shown as struck-through or with special markings, etc.

Both of the following examples correspond to November 5, 2001, 8:15:30 am, US Eastern Standard Time.

Example

```
2001-11-05T13:15:30Z
2001-11-05T08:15:30-05:00
```

Used with ins [p.154] , this gives:

Example

```
<ins datetime="2001-11-05T08:15:30-05:00"
        cite="http://www.example.org/mydoc/comments.html">
Furthermore, the latest figures from the marketing department
suggest that such practice is on the rise.
</ins>
```

The document "http://www.example.org/mydoc/comments.html" would contain comments about why information was inserted into the document.

Authors may also make comments about deleted text by means of the @title [p.62] attribute for the del [p.153] element. User agents may present this information to the user (e.g., as a popup note). For example:

Example

```
<del datetime="2001-11-05T08:15:30-05:00"
        title="Changed as a result of Steve G's comments in meeting.">
Furthermore, the latest figures from the marketing department
suggest that such practice is on the rise.
</del>
```

35.2. The ins element

The ins [p.154] element is used to indicate that a section of a document has been inserted with respect to a different version of the document (e.g., in draft legislation where lawmakers need to view the changes).

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

User agents should render inserted text in ways that make the change obvious. For instance, inserted text may appear in a special font.

36. XHTML Legacy Headings Module

This section is *normative*.

This module defines the traditional HTML heading elements `h1` - `h6`.

Element	Attributes	Content Model
h1 [p.155]	Common [p.34]	(Text [p.47])*
h2 [p.155]	Common [p.34]	(Text [p.47])*
h3 [p.155]	Common [p.34]	(Text [p.47])*
h4 [p.155]	Common [p.34]	(Text [p.47])*
h5 [p.155]	Common [p.34]	(Text [p.47])*
h6 [p.155]	Common [p.34]	(Text [p.47])*

Implementations: RELAX NG [p.204] , XML Schema [p.280]

36.1. The heading elements

A heading element briefly describes the topic of the section it introduces. Heading information may be used by user agents, for example, to construct a table of contents for a document automatically.

Attributes

The Common [p.34] collection
: A collection of other attribute collections, including: Bi-directional [p.79] , Core [p.??] , Edit [p.83] , Embedding [p.85] , Events [p.147] , Forms [p.144] , Hypertext [p.65] , I18N [p.71] , Map [p.89] , Metainformation [p.99] , and Role [p.111]

There are six levels of numbered headings in XHTML with h1 [p.155] as the most important and h6 [p.155] as the least.

Example

```
<body>
<h1>This is a top level heading</h1>
    <p>....</p>
    <h2>This is a second-level heading</h2>
    <p>....</p>
    <h2>This is another second-level heading</h2>
    <p>....</p>
<h1>This is another top level heading</h1>
    <p>....</p>
    <h2>This is another second-level heading</h2>
```

```
    <p>....</p>
        <h3>This is a third-level heading</h3>
        <p>....</p>
</body>
```

Numbered sections and references

XHTML does not itself cause section numbers to be generated from headings. Style sheet languages such as CSS however allow authors to control the generation of section numbers.

Skipping heading levels is considered to be bad practice. The series `h1 h2 h1` *is acceptable, while* `h1 h3 h1` *is not, since the heading level* `h2` *has been skipped.*

A. Changes from earlier XHTML versions

This appendix is *informative*.

This Appendix describes the differences between XHTML 2.0 and XHTML 1.1. It also describes some important ways that changes in XHTML 2.0 may effect creators of RDFa processors [RDFASYNTAX [p.310]].

B. XHTML 2.0 RELAX NG Definition

This appendix is *normative*.

This appendix contains the implementation of the XHTML 2.0 RELAX NG driver file.

B.0.1. RELAX NG XHTML 2.0 Driver

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar ns="http://www.w3.org/1999/xhtml"
         xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>RELAX NG schema for XHTML 2.0</x:h1>
  <x:pre>
    Copyright Â©2003-2009 W3CÂ® (MIT, ERCIM, Keio), All Rights Reserved.
      Editors:   Masayasu Ishikawa &lt;mimasa@w3.org&gt;
                 Markus Gylling &lt;markus.gylling@tpb.se&gt;
      Revision: $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    Permission to use, copy, modify and distribute this RELAX NG schema
    for XHTML 2.0 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 this RELAX NG
    schema for any purpose.
    It is provided "as is" without expressed or implied warranty.
    For details, please refer to the W3C software license at:
      <x:a href="http://www.w3.org/Consortium/Legal/copyright-software"
      >http://www.w3.org/Consortium/Legal/copyright-software</x:a>
  </x:pre>
  <div>
    <x:h2>XHTML 2.0 modules</x:h2>
    <x:h3>Attribute Collections Module</x:h3>
    <include href="xhtml-attribs-2.rng"/>
    <x:h3>Document Module</x:h3>
    <include href="xhtml-document-2.rng"/>
    <x:h3>Structural Module</x:h3>
    <include href="xhtml-structural-2.rng"/>
    <x:h3>Text Module</x:h3>
    <include href="xhtml-text-2.rng"/>
    <x:h3>Hypertext Module</x:h3>
    <include href="xhtml-hypertext-2.rng"/>
    <x:h3>List Module</x:h3>
    <include href="xhtml-list-2.rng"/>
    <x:h3>Image Module</x:h3>
    <include href="xhtml-image-2.rng"/>
    <x:h3>Metainformation Module</x:h3>
    <include href="xhtml-meta-2.rng"/>
    <x:h3>Object Module</x:h3>
    <include href="xhtml-object-2.rng"/>
    <x:h3>Access Module</x:h3>
    <include href="xhtml-access-2.rng"/>
    <x:h3>Style Sheet Module</x:h3>
    <include href="xhtml-style-2.rng"/>
    <x:h3>Tables Module</x:h3>
    <include href="xhtml-table-2.rng"/>
```

```
  <x:h3>Support Modules</x:h3>
  <x:h4>Datatypes Module</x:h4>
  <include href="xhtml-datatypes-2.rng"/>
  <x:h4>Param Module</x:h4>
  <include href="xhtml-param-2.rng"/>
  <x:h4>Caption Module</x:h4>
  <include href="xhtml-caption-2.rng"/>
</div>
<div>
  <x:h2>XHTML 2.0 Legacy modules</x:h2>
  <x:h3>Legacy Edit Module</x:h3>
  <include href="xhtml-legacy-edit-2.rng"/>
  <x:h3>Legacy Line Break Module</x:h3>
  <include href="xhtml-legacy-br-2.rng"/>
  <x:h3>Legacy Headings Module</x:h3>
  <include href="xhtml-legacy-heading-2.rng"/>
</div>
<div>
  <x:h2>XML Events module</x:h2>
      <include href="xhtml-events-2.rng"/>
</div>
<div>
  <x:h2>Ruby module</x:h2>
  <include href="full-ruby-1.rng">
    <define name="Inline.class">
      <notAllowed/>
    </define>
    <define name="NoRuby.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </include>
  <define name="Inline.model">
    <notAllowed/>
  </define>
  <define name="xhtml.Text.class" combine="choice">
    <ref name="ruby"/>
  </define>
</div>
<div>
  <x:h2>XForms module</x:h2>
  <include href="xforms-nons-11.rng">
    <!-- override the xforms schemas any definition
    to circumvent attribute datatype collision error -->
        <define name="xforms.instance.content">
          <ref name="any"/>  <!-- TODO current excludes xhtml ns -->
        </define>
        <define name="xforms.schema.attrib">
          <ref name="any"/>  <!-- TODO xsd ns only? -->
        </define>
        <define name="xforms.schema.content">
          <ref name="any"/> <!-- TODO xsd ns only? -->
        </define>
  </include>
  <define name="xforms.Common.attrib" combine="interleave">
    <!-- this contributes linking and events attributes as well,
    so no need to redefine xforms.Events.attrib and xforms.Linking.attrib -->
    <!-- we have attribute name collisions with resource, encoding and target
```

```
        An option is to contribute only core and i18n into xforms.Common.attrib -->
        <ref name="xhtml.Common.attrib"/>
      </define>
      <define name="xhtml.head.misc" combine="choice">
        <ref name="xforms.model"/>
      </define>
      <define name="xhtml.Misc.class" combine="choice">
        <ref name="xforms.Core.Form.Controls"/>
        <ref name="xforms.Actions"/>
        <ref name="xforms.group"/>
        <ref name="xforms.switch"/>
        <ref name="xforms.repeat"/>
      </define>
      <define name="xforms.UI.Inline.class" combine="choice">
        <ref name="xhtml.Text.class"/>
        <ref name="xhtml.Misc.class"/>
      </define>
      <define name="any">
        <element>
          <anyName>
            <except>
              <nsName ns="http://www.w3.org/1999/xhtml"/>
            </except>
          </anyName>
          <zeroOrMore>
            <attribute>
              <anyName/>
            </attribute>
          </zeroOrMore>
          <zeroOrMore>
            <choice>
              <ref name="any"/>
              <text/>
            </choice>
          </zeroOrMore>
        </element>
      </define>
    </div>
    <div>
      <x:h2>XML Schema instance module</x:h2>
      <include href="XMLSchema-instance.rng"/>
      <define name="xhtml.html.attlist" combine="interleave">
        <optional>
          <ref name="XSI.schemaLocation"/>
        </optional>
      </define>
    </div>
  </grammar>
```

C. XHTML RELAX NG Module Implementations

This appendix is *normative*.

This appendix contains implementations of the modules defined in this specification. These module implementations can be used in other XHTML Family Document Types.

C.1. XHTML Module Implementations

This section contains the formal definition of each of the XHTML Abstract Modules as a RELAX NG module.

C.1.1. Attribute Collections

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Attribute Collections Module</x:h1>
  <div>
    <x:h2>Core Attributes Module</x:h2>
    <include href="xhtml-core-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>Internationalization Attribute Module</x:h2>
    <include href="xhtml-i18n-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>Bi-directional Text Collection</x:h2>
    <include href="xhtml-bidi-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>Edit Attributes Module</x:h2>
    <include href="xhtml-edit-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>Embedding Attributes Module</x:h2>
    <include href="xhtml-embed-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>XForms Repeat Attribute Collection</x:h2>
    <include href="xforms-repeat-attrib.rng"/>
    <define name="xhtml.Common.attrib" combine="interleave">
      <ref name="xforms.Repeat.attrib"/>
    </define>
  </div>
  <div>
    <x:h2>Hypertext Attributes Module</x:h2>
    <include href="xhtml-hypertext-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>Image Map Attributes Module</x:h2>
    <include href="xhtml-imagemap-attrib-2.rng"/>
  </div>
```

```
  <div>
    <x:h2>Media Attribute Module</x:h2>
    <include href="xhtml-media-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>Metainformation Attributes Module</x:h2>
    <include href="xhtml-meta-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>Role Attribute Module</x:h2>
    <include href="xhtml-role-attrib-2.rng"/>
  </div>
  <div>
    <x:h2>Style Attribute Module</x:h2>
    <include href="xhtml-inlstyle-2.rng"/>
  </div>
  <define name="xhtml.Common.extra.attrib">
    <empty/>
  </define>
  <define name="xhtml.Common.attrib">
    <ref name="xhtml.Common.extra.attrib"/>
  </define>
</grammar>
```

C.1.2. Document

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Document Module</x:h1>
  <start>
    <ref name="xhtml.html"/>
  </start>
  <div>
    <x:h2>The html element</x:h2>
    <define name="xhtml.html">
      <element name="html">
        <ref name="xhtml.html.attlist"/>
        <ref name="xhtml.head"/>
        <ref name="xhtml.body"/>
      </element>
    </define>
    <define name="xhtml.html.attlist">
      <ref name="xhtml.Common.attrib"/>
      <optional>
        <ref name="xhtml.version.attrib"/>
      </optional>
    </define>
    <define name="xhtml.version.attrib">
        <attribute name="version">
          <ref name="CDATA.datatype"/>
        </attribute>
    </define>
  </div>
  <div>
    <x:h2>The head element</x:h2>
```

```
    <define name="xhtml.head">
      <element name="head">
        <ref name="xhtml.head.attlist"/>
        <ref name="xhtml.head.content"/>
      </element>
    </define>
    <define name="xhtml.head.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.head.content">
      <ref name="xhtml.title"/>
      <zeroOrMore>
        <choice>
          <ref name="xhtml.head.misc"/>
        </choice>
      </zeroOrMore>
    </define>
    <define name="xhtml.head.misc">
      <notAllowed/>
    </define>
  </div>
  <div>
    <x:h2>The title element</x:h2>
    <define name="xhtml.title">
      <element name="title">
        <ref name="xhtml.title.attlist"/>
        <ref name="xhtml.title.content"/>
      </element>
    </define>
    <define name="xhtml.title.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.title.content">
      <ref name="xhtml.Text.model"/>
    </define>
    <define name="xhtml.Text.class" combine="choice">
      <ref name="xhtml.title"/>
    </define>
  </div>
  <div>
    <x:h2>The body element</x:h2>
    <define name="xhtml.body">
      <element name="body">
        <ref name="xhtml.body.attlist"/>
        <ref name="xhtml.body.content"/>
      </element>
    </define>
    <define name="xhtml.body.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.body.content">
      <ref name="xhtml.Structural.model"/>
    </define>
  </div>
</grammar>
```

C.1.3. Structural

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Structural Module</x:h1>
  <div>
    <x:h2>The address element</x:h2>
    <define name="xhtml.address">
      <element name="address">
        <ref name="xhtml.address.attlist"/>
        <ref name="xhtml.address.content"/>
      </element>
    </define>
    <define name="xhtml.address.attlist">
      <ref name="xhtml.Structural.attrib"/>
    </define>
    <define name="xhtml.address.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The blockcode element</x:h2>
    <define name="xhtml.blockcode">
      <element name="blockcode">
        <ref name="xhtml.blockcode.attlist"/>
        <ref name="xhtml.blockcode.content"/>
      </element>
    </define>
    <define name="xhtml.blockcode.attlist">
      <ref name="xhtml.Structural.attrib"/>
    </define>
    <define name="xhtml.blockcode.content">
      <zeroOrMore>
        <choice>
          <text/>
          <ref name="xhtml.Text.class"/>
          <ref name="xhtml.Heading.class"/>
          <ref name="xhtml.Structural.class"/>
          <ref name="xhtml.List.class"/>
          <ref name="xhtml.Misc.class"/>
        </choice>
      </zeroOrMore>
    </define>
  </div>
  <div>
    <x:h2>The blockquote element</x:h2>
    <define name="xhtml.blockquote">
      <element name="blockquote">
        <ref name="xhtml.blockquote.attlist"/>
        <ref name="xhtml.blockquote.content"/>
      </element>
    </define>
    <define name="xhtml.blockquote.attlist">
      <ref name="xhtml.Structural.attrib"/>
    </define>
```

```
    <define name="xhtml.blockquote.content">
      <zeroOrMore>
        <choice>
          <text/>
          <ref name="xhtml.Text.class"/>
          <ref name="xhtml.Heading.class"/>
          <ref name="xhtml.Structural.class"/>
          <ref name="xhtml.List.class"/>
          <ref name="xhtml.Misc.class"/>
        </choice>
      </zeroOrMore>
    </define>
  </div>
  <div>
    <x:h2>The div element</x:h2>
    <define name="xhtml.div">
      <element name="div">
        <ref name="xhtml.div.attlist"/>
        <ref name="xhtml.div.content"/>
      </element>
    </define>
    <define name="xhtml.div.attlist">
      <ref name="xhtml.Structural.attrib"/>
    </define>
    <define name="xhtml.div.content">
      <ref name="xhtml.Flow.model"/>
    </define>
  </div>
  <div>
    <x:h2>The heading element</x:h2>
    <define name="xhtml.h">
      <element name="h">
        <ref name="xhtml.h.attlist"/>
        <ref name="xhtml.h.content"/>
      </element>
    </define>
    <define name="xhtml.h.attlist">
      <ref name="xhtml.Structural.attrib"/>
    </define>
    <define name="xhtml.h.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The p element</x:h2>
    <define name="xhtml.p">
      <element name="p">
        <ref name="xhtml.p.attlist"/>
        <ref name="xhtml.p.content"/>
      </element>
    </define>
    <define name="xhtml.p.attlist">
      <ref name="xhtml.Structural.attrib"/>
    </define>
    <define name="xhtml.p.content">
      <zeroOrMore>
        <choice>
```

```
        <text/>
        <ref name="xhtml.Text.class"/>
        <ref name="xhtml.List.class"/>
        <ref name="xhtml.blockcode"/>
        <ref name="xhtml.blockquote"/>
        <ref name="xhtml.pre"/>
        <ref name="xhtml.table"/>
        <ref name="xhtml.Misc.class"/>
      </choice>
    </zeroOrMore>
  </define>
</div>
<div>
  <x:h2>The pre element</x:h2>
  <define name="xhtml.pre">
    <element name="pre">
      <ref name="xhtml.pre.attlist"/>
      <ref name="xhtml.pre.content"/>
    </element>
  </define>
  <define name="xhtml.pre.attlist">
    <ref name="xhtml.Structural.attrib"/>
  </define>
  <define name="xhtml.pre.content">
    <ref name="xhtml.Text.model"/>
  </define>
</div>
<div>
  <x:h2>The section element</x:h2>
  <define name="xhtml.section">
    <element name="section">
      <ref name="xhtml.section.attlist"/>
      <ref name="xhtml.section.content"/>
    </element>
  </define>
  <define name="xhtml.section.attlist">
    <ref name="xhtml.Structural.attrib"/>
  </define>
  <define name="xhtml.section.content">
    <ref name="xhtml.Flow.model"/>
  </define>
  <!-- This pattern is not referenced by the default
    XHTML2 content model, but provided for the convenience
    of language designers -->
  <define name="xhtml.section.strict.model">
    <zeroOrMore>
      <choice>
        <ref name="xhtml.section.strict.prehead.extra"/>
      </choice>
    </zeroOrMore>
    <ref name="xhtml.Heading.class" />
        <zeroOrMore>
      <choice>
        <ref name="xhtml.Structural.class"/>
      </choice>
    </zeroOrMore>
  </define>
```

```
    <define name="xhtml.section.strict.prehead.extra">
      <empty/>
    </define>
  </div>
  <div>
    <x:h2>The separator element</x:h2>
    <define name="xhtml.separator">
      <element name="separator">
        <ref name="xhtml.separator.attlist"/>
        <empty/>
      </element>
    </define>
    <define name="xhtml.separator.attlist">
      <ref name="xhtml.Structural.attrib"/>
    </define>
  </div>
  <div>
    <x:h2>Content Model</x:h2>
    <define name="xhtml.Heading.class">
      <choice>
        <ref name="xhtml.h"/>
      </choice>
    </define>
    <define name="xhtml.Structural.nosection.class">
      <choice>
        <ref name="xhtml.address"/>
        <ref name="xhtml.blockcode"/>
        <ref name="xhtml.blockquote"/>
        <ref name="xhtml.div"/>
        <ref name="xhtml.p"/>
        <ref name="xhtml.pre"/>
        <ref name="xhtml.separator"/>
      </choice>
    </define>
    <define name="xhtml.Structural.class">
      <choice>
        <ref name="xhtml.Structural.nosection.class" />
        <ref name="xhtml.section"/>
      </choice>
    </define>
    <define name="xhtml.Structural.mix">
      <zeroOrMore>
        <choice>
          <ref name="xhtml.Heading.class"/>
          <ref name="xhtml.Structural.class"/>
          <ref name="xhtml.List.class"/>
          <ref name="xhtml.Misc.class"/>
        </choice>
      </zeroOrMore>
    </define>
    <define name="xhtml.Structural.model">
      <oneOrMore>
        <ref name="xhtml.Structural.mix"/>
      </oneOrMore>
    </define>
    <define name="xhtml.Flow.model">
      <zeroOrMore>
```

```
          <choice>
            <text/>
            <ref name="xhtml.Heading.class"/>
            <ref name="xhtml.Structural.class"/>
            <ref name="xhtml.List.class"/>
            <ref name="xhtml.Text.class"/>
            <ref name="xhtml.Misc.class"/>
          </choice>
        </zeroOrMore>
      </define>
      <define name="xhtml.Structural.attrib" combine="interleave">
        <ref name="xhtml.Common.attrib" />
      </define>
    </div>
  </grammar>
```

C.1.4. Text

```
  <?xml version="1.0" encoding="UTF-8"?>
  <grammar xmlns="http://relaxng.org/ns/structure/1.0"
           xmlns:x="http://www.w3.org/1999/xhtml">
    <x:h1>Text Module</x:h1>
    <div>
      <x:h2>The abbr element</x:h2>
      <define name="xhtml.abbr">
        <element name="abbr">
          <ref name="xhtml.abbr.attlist"/>
          <ref name="xhtml.abbr.content"/>
        </element>
      </define>
      <define name="xhtml.abbr.attlist">
        <ref name="xhtml.Text.attrib"/>
        <optional>
          <attribute name="full">
            <ref name="URI.datatype"/>
          </attribute>
        </optional>
      </define>
      <define name="xhtml.abbr.content">
        <ref name="xhtml.Text.model"/>
      </define>
    </div>
    <div>
      <x:h2>The cite element</x:h2>
      <define name="xhtml.cite">
        <element name="cite">
          <ref name="xhtml.cite.attlist"/>
          <ref name="xhtml.cite.content"/>
        </element>
      </define>
      <define name="xhtml.cite.attlist">
        <ref name="xhtml.Text.attrib"/>
      </define>
      <define name="xhtml.cite.content">
          <ref name="xhtml.Text.model"/>
      </define>
```

```
</div>
<div>
  <x:h2>The code element</x:h2>
  <define name="xhtml.code">
    <element name="code">
      <ref name="xhtml.code.attlist"/>
      <ref name="xhtml.code.content"/>
    </element>
  </define>
  <define name="xhtml.code.attlist">
    <ref name="xhtml.Text.attrib"/>
  </define>
  <define name="xhtml.code.content">
      <ref name="xhtml.Text.model"/>
  </define>
</div>
<div>
  <x:h2>The dfn element</x:h2>
  <define name="xhtml.dfn">
    <element name="dfn">
      <ref name="xhtml.dfn.attlist"/>
      <ref name="xhtml.dfn.content"/>
    </element>
  </define>
  <define name="xhtml.dfn.attlist">
    <ref name="xhtml.Text.attrib"/>
  </define>
  <define name="xhtml.dfn.content">
    <ref name="xhtml.Text.model"/>
  </define>
</div>
<div>
  <x:h2>The em element</x:h2>
  <define name="xhtml.em">
    <element name="em">
      <ref name="xhtml.em.attlist"/>
      <ref name="xhtml.em.content"/>
    </element>
  </define>
  <define name="xhtml.em.attlist">
    <ref name="xhtml.Text.attrib"/>
  </define>
  <define name="xhtml.em.content">
    <ref name="xhtml.Text.model"/>
  </define>
</div>
<div>
  <x:h2>The kbd element</x:h2>
  <define name="xhtml.kbd">
    <element name="kbd">
      <ref name="xhtml.kbd.attlist"/>
      <ref name="xhtml.kbd.content"/>
    </element>
  </define>
  <define name="xhtml.kbd.attlist">
    <ref name="xhtml.Text.attrib"/>
  </define>
```

```
    <define name="xhtml.kbd.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The l element</x:h2>
    <define name="xhtml.l">
      <element name="l">
        <ref name="xhtml.l.attlist"/>
        <ref name="xhtml.l.content"/>
      </element>
    </define>
    <define name="xhtml.l.attlist">
      <ref name="xhtml.Text.attrib"/>
    </define>
    <define name="xhtml.l.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The quote element</x:h2>
    <define name="xhtml.quote">
      <element name="q">
        <ref name="xhtml.quote.attlist"/>
        <ref name="xhtml.quote.content"/>
      </element>
    </define>
    <define name="xhtml.quote.attlist">
      <ref name="xhtml.Text.attrib"/>
    </define>
    <define name="xhtml.quote.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The samp element</x:h2>
    <define name="xhtml.samp">
      <element name="samp">
        <ref name="xhtml.samp.attlist"/>
        <ref name="xhtml.samp.content"/>
      </element>
    </define>
    <define name="xhtml.samp.attlist">
      <ref name="xhtml.Text.attrib"/>
    </define>
    <define name="xhtml.samp.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The span element</x:h2>
    <define name="xhtml.span">
      <element name="span">
        <ref name="xhtml.span.attlist"/>
        <ref name="xhtml.span.content"/>
      </element>
    </define>
```

```
    <define name="xhtml.span.attlist">
      <ref name="xhtml.Text.attrib"/>
    </define>
    <define name="xhtml.span.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The strong element</x:h2>
    <define name="xhtml.strong">
      <element name="strong">
        <ref name="xhtml.strong.attlist"/>
        <ref name="xhtml.strong.content"/>
      </element>
    </define>
    <define name="xhtml.strong.attlist">
      <ref name="xhtml.Text.attrib"/>
    </define>
    <define name="xhtml.strong.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The sub element</x:h2>
    <define name="xhtml.sub">
      <element name="sub">
        <ref name="xhtml.sub.attlist"/>
        <ref name="xhtml.sub.content"/>
      </element>
    </define>
    <define name="xhtml.sub.attlist">
      <ref name="xhtml.Text.attrib"/>
    </define>
    <define name="xhtml.sub.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The sup element</x:h2>
    <define name="xhtml.sup">
      <element name="sup">
        <ref name="xhtml.sup.attlist"/>
        <ref name="xhtml.sup.content"/>
      </element>
    </define>
    <define name="xhtml.sup.attlist">
      <ref name="xhtml.Text.attrib"/>
    </define>
    <define name="xhtml.sup.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:h2>The var element</x:h2>
    <define name="xhtml.var">
      <element name="var">
        <ref name="xhtml.var.attlist"/>
```

```
        <ref name="xhtml.var.content"/>
      </element>
    </define>
    <define name="xhtml.var.attlist">
      <ref name="xhtml.Text.attrib"/>
    </define>
    <define name="xhtml.var.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <div>
    <x:p>these can occur at Structural or Text level</x:p>
    <define name="xhtml.Misc.class">
      <empty/>
    </define>
  </div>
  <div>
    <x:h2>Content Model</x:h2>
    <define name="xhtml.Text.class">
      <choice>
        <ref name="xhtml.abbr"/>
        <ref name="xhtml.cite"/>
        <ref name="xhtml.code"/>
        <ref name="xhtml.dfn"/>
        <ref name="xhtml.em"/>
        <ref name="xhtml.kbd"/>
        <ref name="xhtml.l"/>
        <ref name="xhtml.quote"/>
        <ref name="xhtml.samp"/>
        <ref name="xhtml.span"/>
        <ref name="xhtml.strong"/>
        <ref name="xhtml.sub"/>
        <ref name="xhtml.sup"/>
        <ref name="xhtml.var"/>
      </choice>
    </define>
    <define name="xhtml.Text.model">
      <zeroOrMore>
        <choice>
          <text/>
          <ref name="xhtml.Text.class"/>
          <ref name="xhtml.Misc.class"/>
        </choice>
      </zeroOrMore>
    </define>
    <define name="xhtml.Text.attrib" combine="interleave">
      <ref name="xhtml.Common.attrib"/>
    </define>
  </div>
</grammar>
```

C.1.5. Hypertext

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Hypertext Module</x:h1>
  <div>
    <x:h2>The a element</x:h2>
    <define name="xhtml.a">
      <element name="a">
        <ref name="xhtml.a.attlist"/>
        <ref name="xhtml.a.content"/>
      </element>
    </define>
    <define name="xhtml.a.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.a.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <define name="xhtml.Text.class" combine="choice">
    <ref name="xhtml.a"/>
  </define>
</grammar>
```

C.1.6. List

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>List Module</x:h1>
  <div>
    <x:h2>The dl element</x:h2>
    <define name="xhtml.dl">
      <element name="dl">
        <ref name="xhtml.dl.attlist"/>
        <ref name="xhtml.dl.content"/>
      </element>
    </define>
    <define name="xhtml.dl.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.dl.content">
      <optional>
        <ref name="xhtml.title"/>
      </optional>
      <optional>
        <ref name="xhtml.caption"/>
      </optional>
      <choice>
        <oneOrMore>
          <choice>
            <ref name="xhtml.dt"/>
            <ref name="xhtml.dd"/>
          </choice>
```

```
      </oneOrMore>
      <oneOrMore>
        <ref name="xhtml.di"/>
      </oneOrMore>
    </choice>
  </define>
  <!-- This pattern is not referenced by the default
  XHTML2 content model, but provided for the convenience
  of language designers -->
  <define name="xhtml.dl.content.di.required">
    <optional>
      <ref name="xhtml.caption"/>
    </optional>
    <oneOrMore>
      <ref name="xhtml.di"/>
    </oneOrMore>
  </define>
</div>
<div>
  <x:h2>The di element</x:h2>
  <define name="xhtml.di">
    <element name="di">
      <ref name="xhtml.di.attlist"/>
      <ref name="xhtml.di.content"/>
    </element>
  </define>
  <define name="xhtml.di.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
  <define name="xhtml.di.content">
    <oneOrMore>
      <ref name="xhtml.dt"/>
    </oneOrMore>
    <zeroOrMore>
      <ref name="xhtml.dd"/>
    </zeroOrMore>
  </define>
</div>
<div>
  <x:h2>The dt element</x:h2>
  <define name="xhtml.dt">
    <element name="dt">
      <ref name="xhtml.dt.attlist"/>
      <ref name="xhtml.dt.content"/>
    </element>
  </define>
  <define name="xhtml.dt.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
  <define name="xhtml.dt.content">
    <ref name="xhtml.Text.model"/>
  </define>
</div>
<div>
  <x:h2>The dd element</x:h2>
  <define name="xhtml.dd">
    <element name="dd">
```

```
      <ref name="xhtml.dd.attlist"/>
      <ref name="xhtml.dd.content"/>
    </element>
  </define>
  <define name="xhtml.dd.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
  <define name="xhtml.dd.content">
    <ref name="xhtml.Flow.model"/>
  </define>
</div>
<div>
  <x:h2>The ol element</x:h2>
  <define name="xhtml.ol">
    <element name="ol">
      <ref name="xhtml.ol.attlist"/>
      <ref name="xhtml.ol.content"/>
    </element>
  </define>
  <define name="xhtml.ol.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
  <define name="xhtml.ol.content">
    <optional>
      <ref name="xhtml.title"/>
    </optional>
    <optional>
      <ref name="xhtml.caption"/>
    </optional>
    <oneOrMore>
      <ref name="xhtml.li-in-ol"/>
    </oneOrMore>
  </define>
</div>
<div>
  <x:h2>The ul element</x:h2>
  <define name="xhtml.ul">
    <element name="ul">
      <ref name="xhtml.ul.attlist"/>
      <ref name="xhtml.ul.content"/>
    </element>
  </define>
  <define name="xhtml.ul.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
  <define name="xhtml.ul.content">
    <optional>
      <ref name="xhtml.title"/>
    </optional>
    <optional>
      <ref name="xhtml.caption"/>
    </optional>
    <oneOrMore>
      <ref name="xhtml.li"/>
    </oneOrMore>
  </define>
</div>
```

```
  <div>
    <x:h2>The li element</x:h2>
    <define name="xhtml.li">
      <element name="li">
        <ref name="xhtml.li.attlist"/>
        <ref name="xhtml.li.content"/>
      </element>
    </define>
    <define name="xhtml.li.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.li.content">
        <ref name="xhtml.Flow.model"/>
    </define>
    <define name="xhtml.li-in-ol">
      <element name="li">
        <ref name="xhtml.li-in-ol.attlist"/>
        <ref name="xhtml.li.content"/>
      </element>
    </define>
    <define name="xhtml.li-in-ol.attlist">
      <ref name="xhtml.li.attlist"/>
      <ref name="xhtml.li-in-ol.value.attrib"/>
    </define>
    <define name="xhtml.li-in-ol.value.attrib">
      <optional>
        <attribute name="value">
          <ref name="Number.datatype"/>
        </attribute>
      </optional>
    </define>
  </div>
  <div>
    <x:h2>List Content Set</x:h2>
    <define name="xhtml.List.class">
      <choice>
        <ref name="xhtml.dl"/>
        <ref name="xhtml.ol"/>
        <ref name="xhtml.ul"/>
      </choice>
    </define>
  </div>
</grammar>
```

C.1.7. Core Attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Core Attributes Module</x:h1>
  <div>
    <x:h2>Core Attribute Collection</x:h2>
    <define name="xhtml.class.attrib">
      <attribute name="class">
        <ref name="NMTOKENS.datatype"/>
      </attribute>
```

```
    </define>
    <define name="xhtml.id.attrib">
      <choice>
        <ref name="id.attrib"/>
        <ref name="xmlid.attrib"/>
      </choice>
    </define>
    <define name="xhtml.layout.attrib">
      <!-- a:defaultValue="irrelevant" -->
      <attribute name="layout">
        <choice>
          <value>irrelevant</value>
          <value>relevant</value>
        </choice>
      </attribute>
    </define>
    <define name="xhtml.title.attrib">
       <attribute name="title">
        <ref name="Text.datatype"/>
       </attribute>
    </define>
    <define name="xhtml.Core.attrib">
      <optional>
        <ref name="xhtml.id.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.class.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.layout.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.title.attrib"/>
      </optional>
    </define>
  </div>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.Core.attrib"/>
  </define>
  <define name="id.attrib">
    <attribute name="id">
      <ref name="ID.datatype"/>
    </attribute>
  </define>
  <define name="xmlid.attrib">
    <attribute name="xml:id">
      <ref name="ID.datatype"/>
    </attribute>
  </define>
</grammar>
```

C.1.8. Hypertext Attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Hypertext Attributes Module</x:h1>
  <div>
    <x:h2>Hypertext Attributes Collection</x:h2>
    <define name="xhtml.cite.attrib">
      <attribute name="cite">
        <ref name="URI.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.href.attrib">
      <attribute name="href">
        <ref name="URI.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.hreflang.attrib">
      <attribute name="hreflang">
        <ref name="LanguageCodes.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.hrefmedia.attrib">
      <attribute name="hrefmedia">
        <ref name="MediaDesc.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.hreftype.attrib">
      <attribute name="hreftype">
        <ref name="ContentTypes.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.nextfocus.attrib">
      <attribute name="nextfocus">
        <ref name="IDREF.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.prevfocus.attrib">
      <attribute name="prevfocus">
        <ref name="IDREF.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.target.attrib">
      <attribute name="target">
        <ref name="HrefTarget.datatype"/>
      </attribute>
    </define>
    <define name="xml.base.attrib">
      <attribute name="xml:base">
        <ref name="URI.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.Hypertext.attrib">
      <optional>
        <ref name="xhtml.cite.attrib"/>
```

```
      </optional>
      <optional>
        <ref name="xhtml.href.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.hreflang.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.hrefmedia.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.hreftype.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.nextfocus.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.prevfocus.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.target.attrib"/>
      </optional>
      <optional>
        <ref name="xml.base.attrib"/>
      </optional>
    </define>
  </div>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.Hypertext.attrib"/>
  </define>
</grammar>
```

C.1.9. I18N Attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>I18N Attribute Module</x:h1>
  <div>
    <x:h2>I18N Attribute Collection</x:h2>
    <define name="xml.lang.attrib">
      <attribute name="xml:lang">
        <ref name="LanguageCode.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.I18n.attrib">
      <optional>
        <ref name="xml.lang.attrib"/>
      </optional>
    </define>
  </div>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.I18n.attrib"/>
  </define>
</grammar>
```

C.1.10. Access

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Access Module</x:h1>
  <div>
    <x:h2>The access element</x:h2>
    <define name="xhtml.access">
      <element name="access">
        <empty />
        <ref name="xhtmlAccess.attlist"/>
      </element>
    </define>
    <define name="xhtmlAccess.attlist">
      <ref name="xhtml.Common.attrib"/>
      <ref name="xhtmlAccess.activate.attrib"/>
      <ref name="xhtmlAccess.key.attrib"/>
      <ref name="xhtmlAccess.order.attrib"/>
      <ref name="xhtmlAccess.targetid.attrib"/>
      <ref name="xhtmlAccess.targetrole.attrib"/>
    </define>
    <define name="xhtmlAccess.activate.attrib">
      <optional>
        <!-- a:defaultValue="false" -->
        <attribute name="activate">
          <choice>
            <value>true</value>
            <value>false</value>
          </choice>
        </attribute>
      </optional>
    </define>
    <define name="xhtmlAccess.key.attrib">
      <optional>
        <attribute name="key">
          <ref name="Characters.datatype"/>
        </attribute>
      </optional>
    </define>
    <define name="xhtmlAccess.order.attrib">
      <optional>
        <!-- a:defaultValue="document" -->
        <attribute name="order">
          <choice>
            <value>document</value>
            <value>list</value>
          </choice>
        </attribute>
      </optional>
    </define>
    <define name="xhtmlAccess.targetid.attrib">
      <optional>
        <attribute name="targetid">
          <ref name="IDREF.datatype"/>
        </attribute>
```

```
      </optional>
    </define>
    <define name="xhtmlAccess.targetrole.attrib">
      <optional>
        <attribute name="targetrole">
          <ref name="CURIEs.datatype"/>
        </attribute>
      </optional>
    </define>
  </div>
  <define name="xhtml.head.misc" combine="choice">
    <ref name="xhtml.access"/>
  </define>
</grammar>
```

C.1.11. Bi-directional Text Attribute

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Bi-directional Text Attribute Module</x:h1>
  <div>
    <x:h2>Bi-directional Text Collection</x:h2>
    <define name="xhtml.dir.attrib">
      <optional>
        <!-- a:defaultValue="ltr" -->
        <attribute name="dir">
          <choice>
            <value>ltr</value>
            <value>rtl</value>
            <value>lro</value>
            <value>rlo</value>
          </choice>
        </attribute>
      </optional>
    </define>
    <define name="xhtml.Bidi.attrib">
      <ref name="xhtml.dir.attrib"/>
    </define>
  </div>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.Bidi.attrib"/>
  </define>
</grammar>
```

C.1.12. Edit Attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Edit Attribute Module</x:h1>
  <div>
    <x:h2>Edit Collection</x:h2>
    <define name="xhtml.edit.attrib">
        <attribute name="edit">
          <choice>
```

```
            <value>inserted</value>
            <value>deleted</value>
            <value>changed</value>
            <value>moved</value>
          </choice>
        </attribute>
    </define>
    <define name="xhtml.datetime.attrib">
        <attribute name="datetime">
          <ref name="Datetime.datatype"/>
        </attribute>
    </define>
    <define name="xhtml.Edit.attrib">
      <optional>
        <ref name="xhtml.edit.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.datetime.attrib"/>
      </optional>
    </define>
  </div>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.Edit.attrib"/>
  </define>
</grammar>
```

C.1.13. Embedding Attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Embedding Attributes Module</x:h1>
  <div>
    <x:h2>Embedding Attributes Collection</x:h2>
    <define name="xhtml.src.attrib">
      <attribute name="src">
        <ref name="URI.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.encoding.attrib">
      <attribute name="encoding">
        <ref name="Encodings.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.srctype.attrib">
      <attribute name="srctype">
        <ref name="ContentTypes.datatype"/>
      </attribute>
    </define>
    <define name="xhtml.Embedding.attrib">
      <optional>
        <ref name="xhtml.src.attrib"/>
      </optional>
      <optional>
        <ref name="xhtml.encoding.attrib"/>
      </optional>
```

```
      <optional>
        <ref name="xhtml.srctype.attrib"/>
      </optional>
    </define>
  </div>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.Embedding.attrib"/>
  </define>
</grammar>
```

C.1.14. Image

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Image Module</x:h1>
  <div>
    <x:h2>The img element</x:h2>
    <define name="xhtml.img">
      <element name="img">
        <ref name="xhtml.img.attlist"/>
        <ref name="xhtml.img.content"/>
      </element>
    </define>
    <define name="xhtml.img.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.img.content">
      <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <define name="xhtml.Text.class" combine="choice">
    <ref name="xhtml.img"/>
  </define>
</grammar>
```

C.1.15. Image Map Attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Image Map Attributes Module</x:h1>
  <div>
    <x:h2>Image Map Attributes Collection</x:h2>
    <define name="xhtml.usemap.attrib">
      <optional>
        <attribute name="usemap">
          <ref name="URI.datatype"/>
        </attribute>
      </optional>
    </define>
    <define name="xhtml.ismap.attrib">
      <optional>
        <attribute name="ismap">
          <value>ismap</value>
        </attribute>
```

```
      </optional>
    </define>
    <define name="xhtml.shape.attrib">
      <optional>
        <attribute name="shape">
          <ref name="Shape.datatype"/>
        </attribute>
      </optional>
    </define>
    <define name="xhtml.coords.attrib">
      <optional>
        <attribute name="coords">
          <ref name="Coordinates.datatype"/>
        </attribute>
      </optional>
    </define>
    <define name="xhtml.Map.attrib">
      <ref name="xhtml.usemap.attrib"/>
      <ref name="xhtml.ismap.attrib"/>
      <ref name="xhtml.shape.attrib"/>
      <ref name="xhtml.coords.attrib"/>
    </define>
  </div>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.Map.attrib"/>
  </define>
</grammar>
```

C.1.16. Media Attribute

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Media Attribute Module</x:h1>
  <define name="xhtml.media.attrib">
    <optional>
      <!-- a:defaultValue="all" -->
      <attribute name="media">
        <ref name="MediaDesc.datatype"/>
      </attribute>
    </optional>
  </define>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.media.attrib"/>
  </define>
</grammar>
```

C.1.17. Metainformation Attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
<x:h1>Metainformation Attributes Module</x:h1>
  <div>
    <x:h2>Metadata Attribute Collection</x:h2>
    <define name="rdfa.about.attrib">
```

```
  <optional>
    <attribute name="about">
      <ref name="URIorSafeCURIE.datatype"/>
    </attribute>
  </optional>
</define>
<define name="rdfa.content.attrib">
  <optional>
    <attribute name="content">
      <ref name="CDATA.datatype"/>
    </attribute>
  </optional>
</define>
<define name="rdfa.datatype.attrib">
  <optional>
    <attribute name="datatype">
      <ref name="CURIE.datatype"/>
    </attribute>
  </optional>
</define>
<define name="rdfa.typeof.attrib">
  <optional>
    <attribute name="typeof">
      <ref name="CURIEs.datatype"/>
    </attribute>
  </optional>
</define>
<define name="rdfa.property.attrib">
  <optional>
    <!-- a:defaultValue="reference" -->
    <attribute name="property">
      <ref name="CURIEs.datatype"/>
    </attribute>
  </optional>
</define>
<define name="rdfa.rel.attrib">
  <optional>
    <attribute name="rel">
      <ref name="CURIEs.datatype"/>
    </attribute>
  </optional>
</define>
<define name="rdfa.resource.attrib">
  <optional>
    <attribute name="resource">
      <ref name="URIorSafeCURIE.datatype"/>
    </attribute>
  </optional>
</define>
<define name="rdfa.rev.attrib">
  <optional>
    <attribute name="rev">
      <ref name="CURIEs.datatype"/>
    </attribute>
  </optional>
</define>
<define name="rdfa.Metadata.attrib">
```

```
      <ref name="rdfa.about.attrib"/>
      <ref name="rdfa.content.attrib"/>
      <ref name="rdfa.datatype.attrib"/>
      <ref name="rdfa.typeof.attrib"/>
      <ref name="rdfa.property.attrib"/>
      <ref name="rdfa.rel.attrib"/>
      <ref name="rdfa.resource.attrib"/>
      <ref name="rdfa.rev.attrib"/>
    </define>
  </div>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="rdfa.Metadata.attrib"/>
  </define>
</grammar>
```

C.1.18. Metainformation

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Metainformation Module</x:h1>
  <div>
    <x:h2>The link element</x:h2>
    <define name="xhtml.link">
      <element name="link">
        <ref name="xhtml.link.attlist"/>
        <ref name="xhtml.link.content"/>
      </element>
    </define>
    <define name="xhtml.link.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.link.content">
      <zeroOrMore>
        <choice>
          <ref name="xhtml.link"/>
          <ref name="xhtml.meta"/>
        </choice>
      </zeroOrMore>
    </define>
  </div>
  <define name="xhtml.head.misc" combine="choice">
    <ref name="xhtml.link"/>
  </define>
  <define name="xhtml.Misc.class" combine="choice">
    <ref name="xhtml.link"/>
  </define>
  <div>
    <x:h2>The meta element</x:h2>
    <define name="xhtml.meta">
      <element name="meta">
        <ref name="xhtml.meta.attlist"/>
        <ref name="xhtml.meta.content"/>
      </element>
    </define>
    <define name="xhtml.meta.attlist">
```

```
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.meta.content">
        <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <define name="xhtml.head.misc" combine="choice">
    <ref name="xhtml.meta"/>
  </define>
  <define name="xhtml.Misc.class" combine="choice">
    <ref name="xhtml.meta"/>
  </define>
</grammar>
```

C.1.19. Object

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Object Module</x:h1>
  <x:p>Note. Also include the Caption Module when this module is used.</x:p>
  <div>
    <x:h2>The object element</x:h2>
    <define name="xhtml.object">
      <element name="object">
        <ref name="xhtml.object.attlist"/>
        <ref name="xhtml.object.content"/>
      </element>
    </define>
    <define name="xhtml.object.attlist">
      <ref name="xhtml.Common.attrib"/>
      <optional>
        <attribute name="archive">
          <ref name="URIs.datatype"/>
        </attribute>
      </optional>
      <optional>
        <attribute name="content-length">
          <ref name="Number.datatype"/>
        </attribute>
      </optional>
      <optional>
        <attribute name="declare">
          <value>declare</value>
        </attribute>
      </optional>
    </define>
    <define name="xhtml.object.content">
      <optional>
        <ref name="xhtml.caption"/>
      </optional>
      <optional>
        <ref name="xhtml.standby"/>
      </optional>
      <zeroOrMore>
        <ref name="xhtml.param"/>
```

```
      </zeroOrMore>
      <ref name="xhtml.Flow.model"/>
    </define>
  </div>
  <div>
    <x:h2>The standby element</x:h2>
    <define name="xhtml.standby">
      <element name="standby">
        <ref name="xhtml.standby.attlist"/>
        <ref name="xhtml.standby.model"/>
      </element>
    </define>
    <define name="xhtml.standby.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.standby.model">
        <ref name="xhtml.Text.model"/>
    </define>
  </div>
  <define name="xhtml.Text.class" combine="choice">
    <ref name="xhtml.object"/>
  </define>
</grammar>
```

C.1.20. Style Attribute

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Style Attribute Module</x:h1>
  <define name="xhtml.style.attrib">
    <optional>
      <attribute name="style"/>
    </optional>
  </define>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.style.attrib"/>
  </define>
</grammar>
```

C.1.21. Style Sheet

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Style Module</x:h1>
  <div>
    <x:h2>The style element</x:h2>
    <define name="xhtml.style">
      <element name="style">
        <ref name="xhtml.style.attlist"/>
        <ref name="xhtml.style.content"/>
      </element>
    </define>
    <define name="xhtml.style.attlist">
      <ref name="xhtml.Common.attrib"/>
```

```
      <optional>
        <attribute name="disabled">
          <value>disabled</value>
        </attribute>
      </optional>
    </define>
    <define name="xhtml.style.content">
      <text/>
    </define>
  </div>
  <define name="xhtml.head.misc" combine="choice">
    <ref name="xhtml.style"/>
  </define>
</grammar>
```

C.1.22. Tables

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Tables Module</x:h1>
  <x:p>Note. Also include the Caption Module when this module is used.</x:p>
  <div>
    <x:h2>The table element</x:h2>
    <define name="xhtml.table">
      <element name="table">
        <ref name="xhtml.table.attlist"/>
        <ref name="xhtml.table.content"/>
      </element>
    </define>
    <define name="xhtml.table.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.table.content">
        <optional>
          <ref name="xhtml.title"/>
        </optional>
        <optional>
          <ref name="xhtml.caption"/>
        </optional>
        <optional>
          <ref name="xhtml.summary"/>
        </optional>
        <choice>
          <zeroOrMore>
            <ref name="xhtml.col"/>
          </zeroOrMore>
          <zeroOrMore>
            <ref name="xhtml.colgroup"/>
          </zeroOrMore>
        </choice>
        <choice>
          <group>
            <optional>
              <ref name="xhtml.thead"/>
            </optional>
```

```
            <optional>
              <ref name="xhtml.tfoot"/>
            </optional>
            <oneOrMore>
              <ref name="xhtml.tbody"/>
            </oneOrMore>
          </group>
          <oneOrMore>
            <ref name="xhtml.tr"/>
          </oneOrMore>
        </choice>
    </define>
  </div>
  <div>
    <x:h2>The summary element</x:h2>
    <define name="xhtml.summary">
      <element name="summary">
        <ref name="xhtml.summary.attlist"/>
        <ref name="xhtml.summary.content"/>
      </element>
    </define>
    <define name="xhtml.summary.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.summary.content">
      <ref name="xhtml.Flow.model"/>
    </define>
  </div>
  <div>
    <x:h2>The col element</x:h2>
    <define name="xhtml.col">
      <element name="col">
        <ref name="xhtml.col.attlist"/>
      </element>
    </define>
    <define name="xhtml.col.attlist">
      <ref name="xhtml.Common.attrib"/>
      <ref name="xhtml.span.attrib"/>
    </define>
  </div>
  <div>
    <x:h2>The colgroup element</x:h2>
    <define name="xhtml.colgroup">
      <element name="colgroup">
        <ref name="xhtml.colgroup.attlist"/>
        <ref name="xhtml.colgroup.content"/>
      </element>
    </define>
    <define name="xhtml.colgroup.attlist">
      <ref name="xhtml.Common.attrib"/>
      <ref name="xhtml.span.attrib"/>
    </define>
    <define name="xhtml.colgroup.content">
      <zeroOrMore>
        <ref name="xhtml.col"/>
      </zeroOrMore>
    </define>
```

```
</div>
<div>
  <x:h2>The thead element</x:h2>
  <define name="xhtml.thead">
    <element name="thead">
      <ref name="xhtml.thead.attlist"/>
      <ref name="xhtml.thead.content"/>
    </element>
  </define>
  <define name="xhtml.thead.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
  <define name="xhtml.thead.content">
    <oneOrMore>
      <ref name="xhtml.tr"/>
    </oneOrMore>
  </define>
</div>
<div>
  <x:h2>The tfoot element</x:h2>
  <define name="xhtml.tfoot">
    <element name="tfoot">
      <ref name="xhtml.tfoot.attlist"/>
      <ref name="xhtml.tfoot.content"/>
    </element>
  </define>
  <define name="xhtml.tfoot.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
  <define name="xhtml.tfoot.content">
    <oneOrMore>
      <ref name="xhtml.tr"/>
    </oneOrMore>
  </define>
</div>
<div>
  <x:h2>The tbody element</x:h2>
  <define name="xhtml.tbody">
    <element name="tbody">
      <ref name="xhtml.tbody.attlist"/>
      <ref name="xhtml.tbody.content"/>
   </element>
  </define>
  <define name="xhtml.tbody.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
  <define name="xhtml.tbody.content">
    <oneOrMore>
      <ref name="xhtml.tr"/>
    </oneOrMore>
  </define>
</div>
<div>
  <x:h2>The tr element</x:h2>
  <define name="xhtml.tr">
    <element name="tr">
      <ref name="xhtml.tr.attlist"/>
```

```
      <ref name="xhtml.tr.content"/>
    </element>
  </define>
  <define name="xhtml.tr.attlist">
    <ref name="xhtml.Common.attrib"/>
  </define>
      <define name="xhtml.tr.content">
    <oneOrMore>
      <choice>
        <ref name="xhtml.th"/>
        <ref name="xhtml.td"/>
      </choice>
    </oneOrMore>
      </define>
</div>
<div>
  <x:h2>The th element</x:h2>
  <define name="xhtml.th">
    <element name="th">
      <ref name="xhtml.th.attlist"/>
      <ref name="xhtml.th.content"/>
    </element>
  </define>
  <define name="xhtml.th.attlist">
    <ref name="xhtml.Cell.attrib"/>
  </define>
  <define name="xhtml.th.content">
    <ref name="xhtml.Flow.model"/>
  </define>
</div>
<div>
  <x:h2>The td element</x:h2>
  <define name="xhtml.td">
    <element name="td">
      <ref name="xhtml.td.attlist"/>
      <ref name="xhtml.td.content"/>
    </element>
  </define>
  <define name="xhtml.td.attlist">
    <ref name="xhtml.Cell.attrib"/>
  </define>
  <define name="xhtml.td.content">
      <ref name="xhtml.Flow.model"/>
  </define>
</div>
<div>
  <x:h2>Attribute definitions</x:h2>
  <define name="xhtml.span.attrib">
    <optional>
      <!-- a:defaultValue="1" -->
      <attribute name="span">
        <ref name="spanNumber.datatype"/>
      </attribute>
    </optional>
  </define>
  <define name="xhtml.Cell.attrib">
    <ref name="xhtml.Common.attrib"/>
```

```
      <optional>
        <attribute name="abbr">
          <ref name="Text.datatype"/>
        </attribute>
      </optional>
      <optional>
        <attribute name="axis"/>
      </optional>
      <optional>
        <!-- a:defaultValue="1" -->
        <attribute name="colspan">
          <ref name="Number.datatype"/>
        </attribute>
      </optional>
      <optional>
        <attribute name="headers">
          <ref name="IDREFS.datatype"/>
        </attribute>
      </optional>
      <optional>
        <!-- a:defaultValue="1" -->
        <attribute name="rowspan">
          <ref name="Number.datatype"/>
        </attribute>
      </optional>
      <ref name="xhtml.scope.attrib"/>
    </define>
    <define name="xhtml.scope.attrib">
      <optional>
        <attribute name="scope">
          <choice>
            <value>row</value>
            <value>col</value>
            <value>rowgroup</value>
            <value>colgroup</value>
          </choice>
        </attribute>
      </optional>
    </define>
  </div>
  <define name="xhtml.Structural.class" combine="choice">
    <ref name="xhtml.table"/>
  </define>
</grammar>
```

C.2. XHTML RELAX NG Support Modules

The modules in this section are elements and attributes of the XHTML RELAX NG implementation that, while hidden from casual users, are important to understand when creating derivative markup languages using the Modularization architecture.

C.2.1. Datatypes

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <x:h1>Datatypes Module</x:h1>
  <div>
    <x:h2>Datatypes defined in XML 1.0</x:h2>
    <define name="CDATA.datatype">
      <text/>
    </define>
    <define name="ID.datatype">
      <data type="ID"/>
    </define>
    <define name="IDREF.datatype">
      <data type="IDREF"/>
    </define>
    <define name="IDREFS.datatype">
      <data type="IDREFS"/>
    </define>
    <define name="NAME.datatype">
      <data type="Name"/>
    </define>
    <define name="NMTOKEN.datatype">
      <data type="NMTOKEN"/>
    </define>
    <define name="NMTOKENS.datatype">
      <data type="NMTOKENS"/>
    </define>
  </div>
  <div>
    <x:h2>CURIE Datatypes</x:h2>
    <define name="CURIE.datatype">
      <x:p>A single curie</x:p>
      <data type="string">
        <param name="pattern">(([\i-[:]][\c-[:]]*)?:)?.+</param>
        <param name="minLength">1</param>
      </data>
    </define>
    <define name="CURIEs.datatype">
      <x:p>A whitespace separated list of CURIEs</x:p>
      <list>
        <oneOrMore>
          <ref name="CURIE.datatype"/>
        </oneOrMore>
      </list>
    </define>
    <define name="SafeCURIE.datatype">
      <x:p>A single safe_curie</x:p>
      <data type="string">
        <param name="pattern">\[(([\i-[:]][\c-[:]]*)?:)?.+\]</param>
        <param name="minLength">3</param>
      </data>
    </define>
    <define name="SafeCURIEs.datatype">
```

```
      <x:p>A whitespace separated list of SafeCURIEs</x:p>
      <list>
        <oneOrMore>
          <ref name="SafeCURIE.datatype"/>
        </oneOrMore>
      </list>
    </define>
    <define name="URIorSafeCURIE.datatype">
      <x:p>A URI or a SafeCURIE (since you need a SafeCURIE
        to disambiguate between a common URI and a CURIE)</x:p>
      <choice>
        <ref name="URI.datatype"/>
        <ref name="CURIE.datatype"/>
      </choice>
    </define>
    <define name="URIorSafeCURIEs.datatype">
      <x:p>A whitespace separated list of URIorSafeCURIEs</x:p>
      <list>
        <oneOrMore>
          <ref name="URIorSafeCURIE.datatype"/>
        </oneOrMore>
      </list>
    </define>
  </div>
  <div>
    <x:h2>Additional Datatypes</x:h2>
    <define name="Character.datatype">
      <x:p>A single character, as per section 2.2 of [XML].</x:p>
      <data type="string">
        <param name="length">1</param>
      </data>
    </define>
    <define name="Characters.datatype">
      <data type="string"/>
    </define>
    <define name="Charset.datatype">
      <x:p>A character encoding, as per [RFC2045]</x:p>
      <text/>
    </define>
    <define name="Encodings.datatype">
      <x:p>A comma-separated list of 'charset's with optional q parameters,
        as defined in section 14.2 of [RFC2616] as the field value of
        the Accept-Charset request header.</x:p>
      <text/>
    </define>
    <define name="ContentType.datatype">
      <x:p>Media type, as per [RFC2045]</x:p>
      <text/>
    </define>
    <define name="ContentTypes.datatype">
      <x:p>A list of media ranges with optional accept parameters,
        as defined in section 14.1 of [RFC2616] as the field value
        of the accept request header.</x:p>
      <text/>
    </define>
    <define name="Coordinates.datatype">
      <x:p>Comma separated list of Lengths used in defining areas.</x:p>
```

```
  <data type="string">
    <param name="pattern">(\d+|\d+(\.\d+)?%)(,\s*(\d+|\d+(\.\d+)?%))*</param>
  </data>
</define>
<define name="Datetime.datatype">
  <x:p>Date and time information, as defined by the type dateTime
    in [XMLSCHEMA].</x:p>
  <data type="dateTime"/>
</define>
<define name="HrefTarget.datatype">
  <x:p>Name used as destination for results of certain actions.</x:p>
  <ref name="NMTOKEN.datatype"/>
</define>
<define name="LanguageCode.datatype">
  <x:p>A language code, as per [RFC3066].</x:p>
  <data type="language"/>
</define>
<define name="LanguageCodes.datatype">
  <x:p>A comma-separated list of language ranges.</x:p>
  <text/>
</define>
<define name="Length.datatype">
  <x:p>The value may be either in pixels or a percentage of the available
    horizontal or vertical space. Thus, the value "50%" means half of
    the available space.</x:p>
  <data type="string">
    <param name="pattern">(\d+|\d+(\.\d+)?%)</param>
  </data>
</define>
<define name="LocationPath.datatype">
  <x:p>A location path as defined in [XPATH].</x:p>
  <text/>
</define>
<define name="MediaDesc.datatype">
  <x:p>A comma-separated list of media descriptors as described by [CSS].
    The default is all.</x:p>
  <data type="string">
    <param name="pattern">[^,]+(,\s*[^,]+)*</param>
  </data>
</define>
<define name="Number.datatype">
  <x:p>One or more digits (NUMBER).</x:p>
  <data type="nonNegativeInteger">
    <param name="pattern">[0-9]+</param>
  </data>
</define>
<define name="spanNumber.datatype">
  <x:p>span: this attribute value must be an integer > 0;
    the default value is 1.</x:p>
  <data type="positiveInteger">
    <param name="pattern">[0-9]+</param>
  </data>
</define>
<define name="QName.datatype">
  <x:p>An [XMLNS]-qualified name.</x:p>
  <data type="QName"/>
</define>
```

```
    <define name="QNames.datatype">
      <x:p>One or more white space separated QName values.</x:p>
      <list>
        <oneOrMore>
          <data type="QName"/>
        </oneOrMore>
      </list>
    </define>
    <define name="prefixedQName.datatype">
      <x:p>An [XMLNS]-qualified name.</x:p>
      <data type="QName">
        <param name="pattern">[\i-[:]][\c-[:]]*:[\i-[:]][\c-[:]]*</param>
      </data>
    </define>
    <define name="Shape.datatype">
      <x:p>The shape of a region.</x:p>
      <choice>
        <value>default</value>
        <value>rect</value>
        <value>circle</value>
        <value>poly</value>
      </choice>
    </define>
    <define name="Text.datatype">
      <x:p>Arbitrary textual data, likely meant to be human-readable.</x:p>
      <text/>
    </define>
    <define name="URI.datatype">
      <x:p>A Uniform Resource Identifier Reference, as defined by the type
        anyURI in [XMLSCHEMA].</x:p>
      <data type="anyURI"/>
    </define>
    <define name="URIs.datatype">
      <x:p>A space-separated list of URIs as defined above.</x:p>
      <list>
        <oneOrMore>
          <ref name="URI.datatype"/>
        </oneOrMore>
      </list>
    </define>
    <define name="Boolean.datatype">
      <choice>
            <value>true</value>
            <value>false</value>
          </choice>
        </define>
  </div>
</grammar>
```

C.2.2. Events

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:ev="http://www.w3.org/2001/xml-events"
         xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
```

```
<x:h1>Events Attribute Collection Module</x:h1>
<x:p>This module imports the XMLEvents 2 modules, and
contextualizes them for the XHTML2 document type.</x:p>
  <x:h2>XHTML Events</x:h2>
  <div>
    <include href="xml-events-2.rng" />
    <define name="xmlEvents.Common.attrib" combine="interleave">
      <ref name="CommonNoEvents.attrib"/>
    </define>
    <define name="xhtml.head.misc" combine="choice">
      <ref name="xmlEvents.listener"/>
    </define>
    <define name="xhtml.Structural.class" combine="choice">
      <ref name="xmlEvents.listener"/>
    </define>
  </div>
  <x:h2>XML Handlers</x:h2>
  <div>
    <include href="xml-handlers-2.rng" />
    <define name="xmlHandlers.Common.attrib" combine="interleave">
      <ref name="CommonNoEvents.attrib"/>
    </define>
    <define name="xhtml.head.misc" combine="choice">
      <ref name="xmlHandlers.action"/>
    </define>
    <define name="xhtml.Structural.class" combine="choice">
      <ref name="xmlHandlers.action"/>
    </define>
  </div>
  <x:h2>XML Scripting</x:h2>
  <div>
    <include href="xml-script-2.rng">
      <define name="xmlScripting.script.attlist">
        <ref name="xmlScripting.implements.attrib"/>
        <ref name="xmlScripting.charset.attrib"/>
        <ref name="xhtml.Common.attrib"/>
      </define>
    </include>
    <define name="xmlHandlers.action.content" combine="choice">
      <ref name="xmlScripting.script"/>
    </define>
    <define name="xhtml.head.misc" combine="choice">
      <ref name="xmlScripting.script"/>
    </define>
    <define name="xhtml.Structural.class" combine="choice">
      <ref name="xmlScripting.script"/>
    </define>
    <define name="xhtml.Text.class" combine="choice">
      <ref name="xmlScripting.script"/>
    </define>
  </div>
<x:h2>Events Global Attributes Collection</x:h2>
<div>
  <define name="Events.attrib">
    <optional>
      <ref name="xmlEvents.event.attrib"/>
    </optional>
```

```
        <optional>
          <ref name="xmlEvents.observer.attrib"/>
        </optional>
        <optional>
          <ref name="xmlEvents.eventTarget.attrib"/>
        </optional>
        <optional>
          <ref name="xmlEvents.function.attrib"/>
        </optional>
        <optional>
          <ref name="xmlEvents.handler.attrib"/>
        </optional>
        <optional>
          <ref name="xmlEvents.phase.attrib"/>
        </optional>
        <optional>
          <ref name="xmlEvents.propagate.attrib"/>
        </optional>
        <optional>
          <ref name="xmlEvents.defaultAction.attrib"/>
        </optional>
      </define>
    </div>
    <define name="xhtml.Common.attrib" combine="interleave">
      <ref name="Events.attrib"/>
    </define>
    <!-- A specialized class to avoid duplicate includes of events attributes.
    This class should be an echo of Common.attrib, minus Events.attrib
    We break the pattern of modules contributing themselves to global
    classes, since this is a local concern.
     -->
    <define name="CommonNoEvents.attrib">
      <ref name="xhtml.Common.extra.attrib"/>
      <ref name="xhtml.Core.attrib"/>
      <ref name="xhtml.Edit.attrib"/>
      <ref name="xhtml.Embedding.attrib"/>
      <ref name="xhtml.Hypertext.attrib"/>
      <ref name="xhtml.I18n.attrib"/>
      <ref name="xhtml.role.attrib"/>
      <ref name="rdfa.Metadata.attrib"/>
      <ref name="xhtml.media.attrib"/>
      <ref name="xhtml.style.attrib"/>
      <ref name="xhtml.Map.attrib"/>
    </define>
  </grammar>
```

C.2.3. Param

```
  <?xml version="1.0" encoding="UTF-8"?>
  <grammar xmlns="http://relaxng.org/ns/structure/1.0"
           xmlns:x="http://www.w3.org/1999/xhtml">
    <x:h1>Param Module</x:h1>
    <div>
      <x:h2>The param element</x:h2>
      <define name="xhtml.param">
        <element name="param">
```

```
        <ref name="xhtml.param.attlist"/>
        <empty/>
      </element>
    </define>
    <define name="xhtml.param.attlist">
      <attribute name="name"/>
      <optional>
        <attribute name="value"/>
      </optional>
      <optional>
        <!-- a:defaultValue="data" -->
        <attribute name="valuetype">
          <choice>
            <value>data</value>
            <value>ref</value>
            <value>object</value>
          </choice>
        </attribute>
      </optional>
      <optional>
        <attribute name="type">
          <ref name="ContentTypes.datatype"/>
        </attribute>
      </optional>
    </define>
  </div>
</grammar>
```

C.2.4. Caption

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Caption Module</x:h1>
  <div>
    <x:h2>The caption element</x:h2>
    <define name="xhtml.caption">
      <element name="caption">
        <ref name="xhtml.caption.attlist"/>
        <ref name="xhtml.caption.content"/>
      </element>
    </define>
    <define name="xhtml.caption.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.caption.content">
      <ref name="xhtml.Text.model"/>
    </define>
    <define name="xhtml.Text.class" combine="choice">
      <ref name="xhtml.caption"/>
    </define>
  </div>
</grammar>
```

C.2.5. Role Attribute

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Role Attribute Module</x:h1>
  <define name="xhtml.role.attrib">
    <optional>
      <attribute name="role">
            <ref name="role.content"/>
      </attribute>
    </optional>
  </define>
  <define name="role.content" combine="choice">
    <ref name="CURIEs.datatype"/>
  </define>
  <define name="xhtml.Common.attrib" combine="interleave">
    <ref name="xhtml.role.attrib"/>
  </define>
</grammar>
```

C.3. XHTML RELAX NG Legacy Modules

C.3.1. Legacy Edit

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
    <x:h1>Legacy Edit Module</x:h1>
    <div>
      <x:h2>The ins element</x:h2>
      <define name="xhtml.ins">
        <element name="ins">
          <ref name="xhtml.ins.attlist"/>
          <ref name="xhtml.ins.content"/>
        </element>
      </define>
      <define name="xhtml.ins.attlist">
        <ref name="xhtml.Common.attrib"/>
      </define>
      <define name="xhtml.ins.content">
        <ref name="xhtml.Text.model"/>
      </define>
    </div>
    <div>
      <x:h2>The del element</x:h2>
      <define name="xhtml.del">
        <element name="del">
          <ref name="xhtml.del.attlist"/>
          <ref name="xhtml.del.content"/>
        </element>
      </define>
      <define name="xhtml.del.attlist">
        <ref name="xhtml.Common.attrib"/>
      </define>
```

```
      <define name="xhtml.del.content">
        <ref name="xhtml.Text.model"/>
      </define>
    </div>
    <define name="xhtml.Text.class" combine="choice">
      <ref name="xhtml.ins"/>
      <ref name="xhtml.del"/>
    </define>
</grammar>
```

C.3.2. Legacy Headings

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
    <x:h1>Legacy Headings Module</x:h1>
    <define name="xhtml.h1">
      <element name="h1">
        <ref name="xhtml.legacyh.attlist"/>
        <ref name="xhtml.legacyh.content"/>
      </element>
    </define>
    <define name="xhtml.h2">
      <element name="h2">
        <ref name="xhtml.legacyh.attlist"/>
        <ref name="xhtml.legacyh.content"/>
      </element>
    </define>
    <define name="xhtml.h3">
      <element name="h3">
        <ref name="xhtml.legacyh.attlist"/>
        <ref name="xhtml.legacyh.content"/>
      </element>
    </define>
    <define name="xhtml.h4">
      <element name="h4">
        <ref name="xhtml.legacyh.attlist"/>
        <ref name="xhtml.legacyh.content"/>
      </element>
    </define>
    <define name="xhtml.h5">
      <element name="h5">
        <ref name="xhtml.legacyh.attlist"/>
        <ref name="xhtml.legacyh.content"/>
      </element>
    </define>
    <define name="xhtml.h6">
      <element name="h6">
        <ref name="xhtml.legacyh.attlist"/>
        <ref name="xhtml.legacyh.content"/>
      </element>
    </define>
    <define name="xhtml.legacyh.attlist">
      <ref name="xhtml.Structural.attrib"/>
    </define>
    <define name="xhtml.legacyh.content">
```

```
      <ref name="xhtml.Text.model"/>
    </define>
    <define name="xhtml.Heading.class" combine="choice">
      <choice>
        <ref name="xhtml.h1"/>
        <ref name="xhtml.h2"/>
        <ref name="xhtml.h3"/>
        <ref name="xhtml.h4"/>
        <ref name="xhtml.h5"/>
        <ref name="xhtml.h6"/>
      </choice>
    </define>
</grammar>
```

C.3.3. Legacy Line Break

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
    <x:h1>Legacy Line Break Module</x:h1>
    <define name="xhtml.br">
      <element name="br">
        <ref name="xhtml.br.attlist"/>
        <ref name="xhtml.br.content"/>
      </element>
    </define>
    <define name="xhtml.br.attlist">
      <ref name="xhtml.Common.attrib"/>
    </define>
    <define name="xhtml.br.content">
      <empty/>
    </define>
    <define name="xhtml.Text.class" combine="choice">
      <ref name="xhtml.br"/>
    </define>
</grammar>
```

C.4. RELAX NG External Modules

These modules are not defined by XHTML, but these definitions are included here for completeness.

C.4.1. Ruby

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <x:h1>Ruby Module in RELAX NG</x:h1>
  <x:pre>
    Ruby Elements
      ruby, rbc, rtc, rb, rt, rp
    This module defines grammars to support ruby annotation markup.
    This module is based on the W3C Ruby Annotation Specification:
```

```
    http://www.w3.org/TR/ruby
  Copyright &#xA9;2003 W3C&#xAE; (MIT, ERCIM, Keio), All Rights Reserved.
    Editor:   Masayasu Ishikawa &lt;mimasa@w3.org&gt;
    Revision: $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
  Permission to use, copy, modify and distribute this RELAX NG schema
  for Ruby Annotation 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 this RELAX NG
  schema for any purpose.
  It is provided "as is" without expressed or implied warranty.
  For details, please refer to the W3C software license at:
    <x:a href="http://www.w3.org/Consortium/Legal/copyright-software"
    >http://www.w3.org/Consortium/Legal/copyright-software</x:a>
</x:pre>
<div>
  <x:h2>patterns for the content model of the ruby element</x:h2>
  <define name="Ruby.content.simple">
    <x:p>Content model of simple ruby</x:p>
    <group>
      <ref name="rb"/>
      <choice>
        <ref name="rt-simple"/>
        <group>
          <ref name="rp"/>
          <ref name="rt-simple"/>
          <ref name="rp"/>
        </group>
      </choice>
    </group>
  </define>
  <define name="Ruby.content.complex">
    <x:p>Content model of complex ruby</x:p>
    <group>
      <ref name="rbc"/>
      <ref name="rtc"/>
      <optional>
        <ref name="rtc"/>
      </optional>
    </group>
  </define>
  <define name="Ruby.content">
    <x:p>Simple ruby is used by default</x:p>
    <ref name="Ruby.content.simple"/>
  </define>
</div>
<div>
  <x:h2>Ruby Elements</x:h2>
  <x:h3>ruby element</x:h3>
  <define name="ruby">
    <element name="ruby">
      <ref name="Ruby.content"/>
      <ref name="Ruby.common.attrib"/>
    </element>
  </define>
  <x:h3>rbc (ruby base component) element</x:h3>
  <define name="rbc">
    <element name="rbc">
      <oneOrMore>
        <ref name="rb"/>
      </oneOrMore>
```

```
      <ref name="Ruby.common.attrib"/>
    </element>
  </define>
  <x:h3>rtc (ruby text component) element</x:h3>
  <define name="rtc">
    <element name="rtc">
      <oneOrMore>
        <ref name="rt-complex"/>
      </oneOrMore>
      <ref name="Ruby.common.attrib"/>
    </element>
  </define>
  <x:h3>rb (ruby base) element</x:h3>
  <define name="rb">
    <element name="rb">
      <ref name="NoRuby.content"/>
      <ref name="Ruby.common.attrib"/>
    </element>
  </define>
  <x:h3>rt (ruby text) element</x:h3>
  <define name="rt-simple">
    <x:p>grammar for simple ruby</x:p>
    <x:p>rbspan attribute is not allowed in simple ruby</x:p>
    <element name="rt">
      <ref name="NoRuby.content"/>
      <ref name="Ruby.common.attrib"/>
    </element>
  </define>
  <define name="rt-complex">
    <x:p>grammar for complex ruby</x:p>
    <element name="rt">
      <ref name="NoRuby.content"/>
      <ref name="Ruby.common.attrib"/>
      <optional>
        <!-- mgylling: remove a:defaultValue="1" on rbspan as it competes with
        the rt-simple pattern. See http://lists.dsdl.org/dsdl-discuss/2008-05/0003.html
        Jing does not catch this, but MSV does.
        -->
        <attribute name="rbspan">
          <data type="positiveInteger">
            <param name="pattern">[1-9][0-9]*</param>
          </data>
        </attribute>
      </optional>
    </element>
  </define>
  <x:h3>rp (ruby parenthesis) element</x:h3>
  <define name="rp">
    <element name="rp">
      <text/>
      <ref name="Ruby.common.attrib"/>
    </element>
  </define>
</div>
<div>
  <x:h2>Ruby Common Attributes</x:h2>
  <x:p>Ruby elements are intended to have common attributes of its
    parent markup language. The pattern "xhtml.Common.attrib" MUST be
    defined to integrate this module.</x:p>
  <define name="Ruby.common.attrib">
    <ref name="xhtml.Common.attrib"/>
```

```
    </define>
  </div>
  <div>
    <x:p>Content models of the rb and the rt elements are intended to
      allow other inline-level elements of its parent markup language,
      but it should not include ruby descendent elements. This RELAX NG
      module itself doesn't check nesting of ruby elements.
      The patterns "Inline.class" and "Inline.model" MUST be defined
      to integrate this module.</x:p>
    <define name="Inline.class" combine="choice">
      <ref name="ruby"/>
    </define>
    <define name="NoRuby.content">
      <ref name="Inline.model"/>
    </define>
  </div>
</grammar>
```

C.4.2. Ruby Driver for Full Ruby Markup

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml">
  <x:h1>Ruby Module in RELAX NG for full ruby markup</x:h1>
  <x:pre>
    Copyright &#xA9;2003 W3C&#xAE; (MIT, ERCIM, Keio), All Rights Reserved.
      Editor:   Masayasu Ishikawa &lt;mimasa@w3.org&gt;
      Revision: $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
  </x:pre>
  <include href="ruby-1.rng"/>
  <define name="Ruby.content" combine="choice">
    <x:p>Allow complex ruby markup in addition to simple ruby markup</x:p>
    <ref name="Ruby.content.complex"/>
  </define>
</grammar>
```

C.4.3. XML Events

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <x:h1>XML Events Module in RELAX NG</x:h1>
  <x:pre>
    This is the RELAX NG Schema for the XML Events 2 XML Events module.
    Copyright &#xA9;2003-2009 W3C&#xAE; (MIT, ERCIM, Keio), All Rights Reserved.
    URI: http://www.w3.org/MarkUp/RELAXNG/xml-events-2.rng
    Editors:   Masayasu Ishikawa &lt;mimasa@w3.org&gt;
               Markus Gylling &lt;markus.gylling@tpb.se&gt;
    Revision: $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    Permission to use, copy, modify and distribute this RELAX NG schema
    for XML Events 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 this RELAX NG
    schema for any purpose.
    It is provided "as is" without expressed or implied warranty.
```

```
  For details, please refer to the W3C software license at:
    <x:a href="http://www.w3.org/Consortium/Legal/copyright-software"
    >http://www.w3.org/Consortium/Legal/copyright-software</x:a>
</x:pre>
<define name="xmlEvents.listener">
  <element name="listener">
    <ref name="xmlEvents.listener.attlist"/>
  </element>
</define>
<define name="xmlEvents.listener.attlist">
      <ref name="xmlEvents.event.attrib"/>
  <optional>
    <ref name="xmlEvents.observer.attrib"/>
  </optional>
  <optional>
    <ref name="xmlEvents.eventTarget.attrib"/>
  </optional>
  <optional>
        <ref name="xmlEvents.function.attrib"/>
      </optional>
  <optional>
    <ref name="xmlEvents.handler.attrib"/>
  </optional>
  <optional>
    <ref name="xmlEvents.phase.attrib"/>
  </optional>
  <optional>
    <ref name="xmlEvents.propagate.attrib"/>
  </optional>
  <optional>
    <ref name="xmlEvents.defaultAction.attrib"/>
  </optional>
  <ref name="xmlEvents.Common.attrib"/>
</define>
<define name="xmlEvents.Common.attrib">
  <!-- The host grammar must contribute an
  attribute of type ID to this class -->
  <empty/>
</define>
<define name="xmlEvents.event.attrib">
  <attribute name="event">
    <list>
      <oneOrMore>
        <data type="QName"/>
      </oneOrMore>
    </list>
  </attribute>
</define>
<define name="xmlEvents.eventTarget.attrib">
  <attribute name="eventTarget">
    <data type="IDREFS"/>
  </attribute>
</define>
<define name="xmlEvents.phase.attrib">
  <!-- a:defaultValue="default" -->
  <attribute name="phase">
    <choice>
```

```
        <value>bubble</value>
        <value>capture</value>
        <value>target</value>
        <value>default</value>
      </choice>
    </attribute>
  </define>
  <define name="xmlEvents.handler.attrib">
    <attribute name="handler">
      <data type="anyURI"/>
     </attribute>
  </define>
  <define name="xmlEvents.observer.attrib">
    <attribute name="observer">
      <data type="IDREFS"/>
    </attribute>
  </define>
  <define name="xmlEvents.function.attrib">
      <attribute name="function">
        <text/>
      </attribute>
  </define>
  <define name="xmlEvents.propagate.attrib">
    <!-- a:defaultValue="continue" -->
    <attribute name="propagate">
      <choice>
        <value>stop</value>
        <value>continue</value>
      </choice>
    </attribute>
  </define>
  <define name="xmlEvents.defaultAction.attrib">
    <!-- a:defaultValue="perform" -->
    <attribute name="defaultAction">
      <choice>
        <value>cancel</value>
        <value>perform</value>
      </choice>
    </attribute>
  </define>
</grammar>
```

C.4.4. XML Handlers

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <x:h1>XML Handlers Module in RELAX NG</x:h1>
  <x:pre>
    This is the RELAX NG Schema for the XML Events 2 XML Handlers module.
    As per the XMLEvents 2 specification, this modules depends
    on the XML Events Module (xml-events-2.rng).
    Copyright &#xA9;2009 W3C&#xAE; (MIT, ERCIM, Keio), All Rights Reserved.
    URI: http://www.w3.org/MarkUp/RELAXNG/xml-handlers-2.rng
    Editor:   Markus Gylling &lt;markus.gylling@tpb.se&gt;
    Revision: $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    Permission to use, copy, modify and distribute this RELAX NG schema
    for XML Handlers 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 this RELAX NG
  schema for any purpose.
  It is provided "as is" without expressed or implied warranty.
  For details, please refer to the W3C software license at:
    <x:a href="http://www.w3.org/Consortium/Legal/copyright-software"
    >http://www.w3.org/Consortium/Legal/copyright-software</x:a>
</x:pre>
<define name="xmlHandlers.action">
      <element name="action">
              <ref name="xmlHandlers.action.attlist"/>
              <ref name="xmlHandlers.action.content"/>
      </element>
</define>
<define name="xmlHandlers.action.attlist">
  <ref name="xmlEvents.event.attrib"/>
  <optional>
    <ref name="xmlEvents.eventTarget.attrib"/>
  </optional>
  <ref name="xmlHandlers.declare.attrib"/>
  <ref name="xmlHandlers.if.attrib"/>
  <ref name="xmlHandlers.while.attrib"/>
  <ref name="xmlHandlers.Common.attrib"/>
</define>
<define name="xmlHandlers.action.content">
      <choice>
        <ref name="xmlHandlers.action"/>
        <ref name="xmlHandlers.dispatchEvent"/>
        <ref name="xmlHandlers.addEventListener"/>
        <ref name="xmlHandlers.removeEventListener"/>
        <ref name="xmlHandlers.stopPropagation"/>
        <ref name="xmlHandlers.preventDefault"/>
      </choice>
</define>
<define name="xmlHandlers.dispatchEvent">
      <element name="dispatchEvent">
    <ref name="xmlHandlers.dispatchEvent.attlist"/>
        <empty/>
      </element>
</define>
<define name="xmlHandlers.dispatchEvent.attlist">
  <ref name="xmlHandlers.Common.attrib"/>
  <ref name="xmlHandlers.eventType.attrib"/>
  <ref name="xmlEvents.eventTarget.attrib"/>
  <ref name="xmlHandlers.bubbles.attrib"/>
  <ref name="xmlHandlers.cancelable.attrib"/>
</define>
<define name="xmlHandlers.addEventListener">
      <element name="addEventListener">
    <ref name="xmlHandlers.addEventListener.attlist"/>
        <empty/>
      </element>
</define>
<define name="xmlHandlers.addEventListener.attlist">
      <ref name="xmlHandlers.removeEventListener.attlist"/>
</define>
<define name="xmlHandlers.removeEventListener">
      <element name="removeEventListener">
    <ref name="xmlHandlers.removeEventListener.attlist"/>
        <empty/>
      </element>
</define>
<define name="xmlHandlers.removeEventListener.attlist">
  <ref name="xmlHandlers.event.attrib"/>
  <ref name="xmlEvents.handler.attrib"/>
  <optional>
    <ref name="xmlEvents.phase.attrib"/>
  </optional>
  <ref name="xmlHandlers.Common.attrib"/>
</define>
<define name="xmlHandlers.stopPropagation">
      <element name="stopPropagation">
        <ref name="xmlHandlers.stopPropagation.attlist"/>
        <empty/>
      </element>
</define>
<define name="xmlHandlers.stopPropagation.attlist">
```

```
    <ref name="xmlHandlers.Common.attrib"/>
    <ref name="xmlHandlers.event.attrib"/>
  </define>
  <define name="xmlHandlers.preventDefault">
        <element name="preventDefault">
      <ref name="xmlHandlers.preventDefault.attlist"/>
          <empty/>
        </element>
  </define>
  <define name="xmlHandlers.preventDefault.attlist">
    <ref name="xmlHandlers.Common.attrib"/>
    <ref name="xmlHandlers.event.attrib"/>
  </define>
  <define name="xmlHandlers.Common.attrib">
    <!-- The host grammar must contribute an
    attribute of type ID to this class -->
    <empty/>
  </define>
  <define name="xmlHandlers.event.attrib">
    <attribute name="event">
      <data type="QName"/>
    </attribute>
  </define>
  <define name="xmlHandlers.declare.attrib">
    <optional>
      <attribute name="declare">
        <value>declare</value>
      </attribute>
    </optional>
  </define>
  <define name="xmlHandlers.if.attrib">
    <optional>
      <attribute name="if">
        <data type="normalizedString"/>
      </attribute>
    </optional>
  </define>
  <define name="xmlHandlers.while.attrib">
    <optional>
      <attribute name="while">
        <data type="normalizedString"/>
      </attribute>
    </optional>
  </define>
  <define name="xmlHandlers.eventType.attrib">
    <attribute name="eventType">
        <data type="QName"/>
    </attribute>
  </define>
  <define name="xmlHandlers.bubbles.attrib">
    <optional>
      <attribute name="bubbles">
        <value>bubbles</value>
      </attribute>
    </optional>
  </define>
  <define name="xmlHandlers.cancelable.attrib">
    <optional>
      <attribute name="cancelable">
        <value>cancelable</value>
      </attribute>
    </optional>
  </define>
</grammar>
```

C.4.5. XML Script

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:x="http://www.w3.org/1999/xhtml"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <x:h1>XML Scripting Module in RELAX NG</x:h1>
```

```
<x:p>This module depends on xhtml-datatypes-2.rng</x:p>
<x:pre>
  This is the RELAX NG Schema for the XML Events 2 Scripting module.
  Copyright &#xA9;2009 W3C&#xAE; (MIT, ERCIM, Keio), All Rights Reserved.
  URI: http://www.w3.org/MarkUp/RELAXNG/xml-script-2.rng
  Editor:   Markus Gylling &lt;markus.gylling@tpb.se&gt;
  Revision: $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
  Permission to use, copy, modify and distribute this RELAX NG schema
  for XML Scripting 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 this RELAX NG
  schema for any purpose.
  It is provided "as is" without expressed or implied warranty.
  For details, please refer to the W3C software license at:
  <x:a href="http://www.w3.org/Consortium/Legal/copyright-software"
  >http://www.w3.org/Consortium/Legal/copyright-software</x:a>
</x:pre>
<define name="xmlScripting.script">
  <element name="script">
    <ref name="xmlScripting.script.attlist"/>
    <ref name="xmlScripting.script.model"/>
  </element>
</define>
<define name="xmlScripting.script.attlist">
  <ref name="xmlScripting.encoding.attrib"/>
  <ref name="xmlScripting.charset.attrib"/>
  <ref name="xmlScripting.defer.attrib"/>
  <ref name="xmlScripting.implements.attrib"/>
  <ref name="xmlScripting.src.attrib"/>
  <ref name="xmlScripting.type.attrib"/>
  <ref name="xmlScripting.Common.attrib"/>
</define>
<define name="xmlScripting.script.model">
      <text/>
</define>
<define name="xmlScripting.encoding.attrib">
  <optional>
    <attribute name="encoding">
      <ref name="Encodings.datatype"/>
    </attribute>
  </optional>
</define>
<define name="xmlScripting.charset.attrib">
  <optional>
    <attribute name="charset">
          <ref name="Charset.datatype"/>
    </attribute>
  </optional>
</define>
<define name="xmlScripting.defer.attrib">
  <optional>
    <attribute name="defer">
      <value>defer</value>
    </attribute>
  </optional>
</define>
```

```
  <define name="xmlScripting.Common.attrib">
    <!-- The host grammar must contribute an
    attribute of type ID to this class -->
    <empty/>
  </define>
  <define name="xmlScripting.implements.attrib">
    <optional>
      <attribute name="implements">
        <ref name="URIorSafeCURIEs.datatype"/>
      </attribute>
    </optional>
  </define>
  <define name="xmlScripting.src.attrib">
    <optional>
      <attribute name="src">
        <ref name="URI.datatype"/>
      </attribute>
    </optional>
  </define>
   <define name="xmlScripting.type.attrib">
    <attribute name="type">
      <ref name="ContentType.datatype"/>
    </attribute>
  </define>
</grammar>
```

C.4.6. XML Schema instance

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar xmlns="http://relaxng.org/ns/structure/1.0"
         ns="http://www.w3.org/2001/XMLSchema-instance"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <define name="XSI.type">
    <attribute name="xsi:type">
      <data type="QName"/>
    </attribute>
  </define>
  <define name="XSI.nil">
    <attribute name="xsi:nil">
      <data type="boolean"/>
    </attribute>
  </define>
  <define name="XSI.schemaLocation">
    <attribute name="xsi:schemaLocation">
      <list>
        <oneOrMore>
          <data type="anyURI"/>
          <data type="anyURI"/>
        </oneOrMore>
      </list>
    </attribute>
  </define>
  <define name="XSI.noNamespaceSchemaLocation">
    <attribute name="xsi:noNamespaceSchemaLocation">
```

```
      <data type="anyURI"/>
    </attribute>
  </define>
</grammar>
```

C.4.7. XForms 1.1

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
   Copyright 2004-2005 Brain Attic, L.L.C.
   Licensed under the Apache License, Version 2.0 (the "License");
   you may not use this file except in compliance with the License.
   You may obtain a copy of the License at
       http://www.apache.org/licenses/LICENSE-2.0
   Unless required by applicable law or agreed to in writing, software
   distributed under the License is distributed on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
   See the License for the specific language governing permissions and
   limitations under the License.
-->
<!--
  Draft version of for XForms 1.1 by Leigh L. Klotz, Jr. <Leigh.Klotz@Xerox.com>
  Changes from version of http://lists.w3.org/Archives/Public/public-forms/2008Jun/0023.html
  + added mediatype to output
  + added attribute context to Nodeset.Binding.attrib
  + added attribute context { XPathExpression }? to Single.Node.Binding.attrib
  Changes for integration with XHTML2 by Markus Gylling <markus.gylling@gmail.com>
  + remove start element
  + make chameleon (remove xforms ns)
  + use prefixed define names
  TODO submission doesnt have @target?
-->
<grammar
       xmlns:xsd="http://www.w3.org/2001/XMLSchema"
       xmlns="http://relaxng.org/ns/structure/1.0"
       datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">
  <define name="xforms.Common.attrib">
    <!-- host language to add common attributes here -->
    <empty/>
  </define>
  <define name="xforms.Events.attrib">
    <!-- host language to add XML Events attributes here -->
    <empty/>
  </define>
  <define name="xforms.Linking.attrib">
    <!-- host language to add src attribute here -->
    <empty/>
  </define>
  <define name="xforms.Single.Node.Binding.attrib">
    <choice>
      <attribute name="bind">
        <data type="IDREF"/>
      </attribute>
      <group>
        <optional>
          <attribute name="model">
            <data type="IDREF"/>
          </attribute>
        </optional>
        <attribute name="ref">
          <ref name="XPathExpression"/>
```

```
      </attribute>
      <optional>
        <attribute name="context">
          <ref name="XPathExpression"/>
        </attribute>
      </optional>
    </group>
  </choice>
</define>
<define name="xforms.Nodeset.Binding.attrib">
  <choice>
    <attribute name="bind">
      <data type="IDREF"/>
    </attribute>
    <group>
      <optional>
        <attribute name="model">
          <data type="IDREF"/>
        </attribute>
      </optional>
      <attribute name="nodeset">
        <ref name="XPathExpression"/>
      </attribute>
      <optional>
        <attribute name="context">
          <ref name="XPathExpression"/>
        </attribute>
      </optional>
    </group>
  </choice>
</define>
<!-- Placeholder for XML Schema RNG -->
<define name="xforms.schema">
  <element name="xsd:schema">
    <ref name="xforms.schema.attrib"/>
    <ref name="xforms.schema.content"/>
  </element>
</define>
<define name="xforms.schema.attrib">
      <notAllowed/>
</define>
<define name="xforms.schema.content">
  <notAllowed/>
</define>
<define name="xforms.model">
  <element name="model">
    <ref name="xforms.model.attrib"/>
    <ref name="xforms.model.content"/>
  </element>
</define>
<define name="xforms.model.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.actions.attrib"/>
  <optional>
    <attribute name="functions">
      <list>
        <oneOrMore>
          <data type="QName"/>
        </oneOrMore>
      </list>
    </attribute>
  </optional>
  <optional>
```

```
    <attribute name="schema">
      <list>
        <oneOrMore>
          <data type="anyURI"/>
        </oneOrMore>
      </list>
    </attribute>
  </optional>
  <optional>
    <attribute name="version">
      <list>
        <data type="string">
          <param name="pattern">[1-9]\d*\.\d+</param>
        </data>
      </list>
    </attribute>
  </optional>
</define>
<define name="xforms.model.content">
  <interleave>
    <zeroOrMore>
      <ref name="xforms.instance"/>
    </zeroOrMore>
    <zeroOrMore>
      <ref name="xforms.submission"/>
    </zeroOrMore>
    <zeroOrMore>
      <ref name="xforms.bind"/>
    </zeroOrMore>
    <zeroOrMore>
      <ref name="xforms.schema"/>
    </zeroOrMore>
    <zeroOrMore>
      <ref name="xforms.Actions"/>
    </zeroOrMore>
  </interleave>
</define>
<define name="xforms.instance">
  <element name="instance">
    <ref name="xforms.instance.attrib"/>
    <ref name="xforms.instance.content"/>
  </element>
</define>
<define name="xforms.instance.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Linking.attrib"/>
  <!--TODO collides with rdf/a resource attribute
  <optional>
    <attribute name="resource">
      <data type="anyURI"/>
    </attribute>
  </optional>
  -->
</define>
<define name="xforms.instance.content">
  <optional>
    <ref name="anyElement"/>
  </optional>
</define>
<!--
  TODO: Split of Submission.attrib and Submission.Elements isn't right
  because of OR in atttribute and element cases, so interleave is needed
  to combine them, but we use sequence.
```

```
-->
<define name="xforms.submission">
  <element name="submission">
    <ref name="xforms.submission.model"/>
  </element>
</define>
<define name="xforms.submission.model" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <choice>
    <attribute name="method">
      <choice>
        <ref name="QNameButNotNCName"/>
        <value>post</value>
        <value>put</value>
        <value>get</value>
        <value>delete</value>
        <value>post</value>
        <value>form-data-post</value>
        <value>urlencoded-post</value>
      </choice>
    </attribute>
    <element name="method">
      <ref name="xforms.ValueTemplate"/>
    </element>
  </choice>
  <optional>
    <choice>
      <attribute name="bind">
        <data type="IDREF"/>
      </attribute>
      <attribute name="ref">
        <ref name="XPathExpression"/>
      </attribute>
    </choice>
  </optional>
  <choice>
    <attribute name="action">
      <data type="anyURI"/>
    </attribute>
    <!-- TODO collides with rdf/a resource attribute
    <attribute name="resource">
      <data type="anyURI"/>
    </attribute>
    -->
    <element name="resource">
      <ref name="xforms.ValueTemplate"/>
    </element>
  </choice>
  <optional>
    <element name="header">
      <interleave>
        <element name="name">
          <ref name="xforms.ValueTemplate"/>
        </element>
        <element name="value">
          <ref name="xforms.ValueTemplate"/>
        </element>
      </interleave>
    </element>
  </optional>
  <optional>
    <attribute name="includenamespaceprefixes">
      <list>
```

```
      <choice>
        <data type="NCName"/>
        <value>#default</value>
      </choice>
    </list>
  </attribute>
</optional>
<optional>
  <attribute name="indent">
    <data type="boolean"/>
  </attribute>
</optional>
<!-- TODO collides with xhtml common encoding attribute
<optional>
  <attribute name="encoding"/>
</optional>
-->
<optional>
  <attribute name="mediatype"/>
</optional>
<optional>
  <attribute name="mode">
    <choice>
      <value>asynchronous</value>
      <value>synchronous</value>
    </choice>
  </attribute>
</optional>
<optional>
  <attribute name="omit-xml-declaration">
    <data type="boolean"/>
  </attribute>
</optional>
<optional>
  <attribute name="standalone">
    <data type="boolean"/>
  </attribute>
</optional>
<optional>
  <attribute name="cdata-section-elements">
    <list>
      <oneOrMore>
        <data type="QName"/>
      </oneOrMore>
    </list>
  </attribute>
</optional>
<optional>
  <attribute name="replace">
    <choice>
      <ref name="QNameButNotNCName"/>
      <value>all</value>
      <value>instance</value>
      <value>none</value>
    </choice>
  </attribute>
</optional>
<optional>
  <attribute name="instance">
    <data type="IDREF"/>
  </attribute>
</optional>
<optional>
```

```
    <attribute name="relevant">
      <data type="boolean"/>
    </attribute>
  </optional>
  <optional>
    <attribute name="separator">
      <choice>
        <value>;</value>
        <value>&amp;</value>
      </choice>
    </attribute>
  </optional>
  <optional>
    <attribute name="serialization">
      <choice>
        <value>application/xml</value>
        <value>application/x-www-form-urlencoded</value>
        <value>multipart/related</value>
        <value>multipart/form-data</value>
        <value>none</value>
      </choice>
    </attribute>
  </optional>
  <optional>
    <attribute name="validate">
      <data type="boolean"/>
    </attribute>
  </optional>
  <optional>
    <attribute name="version">
      <data type="NMTOKEN"/>
    </attribute>
  </optional>
</define>
<!--There should only be zero or one resource, zero or one method, and zero or more header
<define name="xforms.submission.content">
  <choice>
    <ref name="xforms.resource"/>
    <ref name="xforms.method"/>
  </choice>
  <optional>
        <oneOrMore>
              <ref name="xforms.header"/>
        </oneOrMore>
  </optional>
   As is the case everywhere else, action handlers are last
  <ref name="xforms.Actions"/>
</define>
-->
<define name="xforms.bind">
  <element name="bind">
    <ref name="xforms.bind.attrib"/>
    <ref name="xforms.bind.content"/>
  </element>
</define>
<define name="xforms.bind.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <attribute name="nodeset">
      <ref name="XPathExpression"/>
    </attribute>
  </optional>
  <optional>
```

```
    <attribute name="type">
      <data type="QName"/>
    </attribute>
  </optional>
  <optional>
    <attribute name="readonly">
      <ref name="XPathExpression"/>
    </attribute>
  </optional>
  <optional>
    <attribute name="required">
      <ref name="XPathExpression"/>
    </attribute>
  </optional>
  <optional>
    <attribute name="relevant">
      <ref name="XPathExpression"/>
    </attribute>
  </optional>
  <optional>
    <attribute name="constraint">
      <ref name="XPathExpression"/>
    </attribute>
  </optional>
  <optional>
    <attribute name="calculate">
      <ref name="XPathExpression"/>
    </attribute>
  </optional>
  <optional>
    <attribute name="p3ptype"/>
  </optional>
</define>
<define name="xforms.bind.content">
  <zeroOrMore>
    <ref name="xforms.bind"/>
  </zeroOrMore>
</define>
<!-- Form Controls -->
<define name="xforms.Core.Form.Controls">
  <choice>
    <ref name="xforms.input"/>
    <ref name="xforms.textarea"/>
    <ref name="xforms.secret"/>
    <ref name="xforms.output"/>
    <ref name="xforms.upload"/>
    <ref name="xforms.select1"/>
    <ref name="xforms.select"/>
    <ref name="xforms.range"/>
    <ref name="xforms.submit"/>
    <ref name="xforms.trigger"/>
  </choice>
</define>
<define name="UI.xforms.Common.attrib" combine="interleave">
  <optional>
    <!-- host language to add accesskey and navindex here -->
    <attribute name="appearance">
      <choice>
        <data type="QName">
          <param name="pattern">[^:]+:[^:]+</param>
        </data>
        <value>minimal</value>
        <value>compact</value>
```

```
        <value>full</value>
      </choice>
    </attribute>
  </optional>
</define>
<define name="xforms.UI.Inline.class">
  <choice>
    <ref name="xforms.output"/>
  </choice>
</define>
<define name="xforms.UI.Inline.content">
  <interleave>
    <text/>
    <optional>
      <ref name="xforms.UI.Inline.class"/>
    </optional>
  </interleave>
</define>
<define name="xforms.UI.Common.content">
  <zeroOrMore>
    <choice>
      <ref name="xforms.help"/>
      <ref name="xforms.hint"/>
      <ref name="xforms.alert"/>
      <ref name="xforms.Actions"/>
    </choice>
  </zeroOrMore>
</define>
<define name="xforms.List.UI.Common.content">
  <oneOrMore>
    <choice>
      <ref name="xforms.item"/>
      <ref name="xforms.itemset"/>
      <ref name="xforms.choices"/>
    </choice>
  </oneOrMore>
</define>
<define name="xforms.label.content">
  <ref name="xforms.UI.Inline.content"/>
</define>
<define name="xforms.label">
  <element name="label">
    <interleave>
      <ref name="xforms.Common.attrib"/>
      <choice>
        <ref name="xforms.Linking.attrib"/>
        <ref name="xforms.Single.Node.Binding.attrib"/>
        <ref name="xforms.label.content"/>
      </choice>
    </interleave>
  </element>
</define>
<define name="xforms.help">
  <element name="help">
    <ref name="xforms.help.attrib"/>
    <ref name="xforms.help.content"/>
  </element>
</define>
<define name="xforms.help.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <ref name="xforms.Linking.attrib"/>
  </optional>
```

```
    <optional>
      <ref name="xforms.Single.Node.Binding.attrib"/>
    </optional>
  </define>
  <define name="xforms.help.content">
    <ref name="xforms.UI.Inline.content"/>
  </define>
  <define name="xforms.hint">
    <element name="hint">
      <ref name="xforms.hint.attrib"/>
      <ref name="xforms.hint.content"/>
    </element>
  </define>
  <define name="xforms.hint.attrib" combine="interleave">
    <ref name="xforms.Common.attrib"/>
    <optional>
      <ref name="xforms.Linking.attrib"/>
    </optional>
    <optional>
      <ref name="xforms.Single.Node.Binding.attrib"/>
    </optional>
  </define>
  <define name="xforms.hint.content">
    <ref name="xforms.UI.Inline.content"/>
  </define>
  <define name="xforms.alert">
    <element name="alert">
      <ref name="xforms.alert.attrib"/>
      <ref name="xforms.alert.content"/>
    </element>
  </define>
  <define name="xforms.alert.attrib" combine="interleave">
    <ref name="xforms.Common.attrib"/>
    <optional>
      <ref name="xforms.Linking.attrib"/>
    </optional>
    <optional>
      <ref name="xforms.Single.Node.Binding.attrib"/>
    </optional>
  </define>
  <define name="xforms.alert.content">
    <ref name="xforms.UI.Inline.content"/>
  </define>
  <define name="xforms.choices">
    <element name="choices">
      <ref name="xforms.choices.attrib"/>
      <ref name="xforms.choices.content"/>
    </element>
  </define>
  <define name="xforms.choices.attrib" combine="interleave">
    <ref name="xforms.Common.attrib"/>
  </define>
  <define name="xforms.choices.content">
    <optional>
      <ref name="xforms.label"/>
    </optional>
    <oneOrMore>
      <choice>
        <ref name="xforms.choices"/>
        <ref name="xforms.item"/>
        <ref name="xforms.itemset"/>
      </choice>
    </oneOrMore>
```

```
</define>
<define name="xforms.value">
  <element name="value">
    <ref name="xforms.value.attrib"/>
    <ref name="xforms.value.content"/>
  </element>
</define>
<define name="xforms.value.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <ref name="xforms.Single.Node.Binding.attrib"/>
  </optional>
</define>
<define name="xforms.value.content">
  <text/>
</define>
<define name="xforms.item">
  <element name="item">
    <ref name="xforms.item.attrib"/>
    <ref name="xforms.item.content"/>
  </element>
</define>
<define name="xforms.item.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
</define>
<define name="xforms.item.content">
  <ref name="xforms.label"/>
  <ref name="xforms.value"/>
  <ref name="xforms.UI.Common.content"/>
</define>
<define name="xforms.itemset">
  <element name="itemset">
    <ref name="xforms.itemset.attrib"/>
    <ref name="xforms.itemset.content"/>
  </element>
</define>
<define name="xforms.itemset.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Nodeset.Binding.attrib"/>
</define>
<define name="xforms.itemset.content">
  <ref name="xforms.label"/>
  <choice>
    <ref name="xforms.value"/>
    <ref name="xforms.copy"/>
  </choice>
  <ref name="xforms.UI.Common.content"/>
</define>
<define name="xforms.copy">
  <element name="copy">
    <ref name="xforms.copy.attrib"/>
    <ref name="xforms.copy.content"/>
  </element>
</define>
<define name="xforms.copy.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Single.Node.Binding.attrib"/>
</define>
<define name="xforms.copy.content">
  <empty/>
</define>
<define name="xforms.filename">
  <element name="filename">
```

```
    <ref name="xforms.filename.attrib"/>
    <ref name="xforms.filename.content"/>
  </element>
</define>
<define name="xforms.filename.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Single.Node.Binding.attrib"/>
</define>
<define name="xforms.filename.content">
  <empty/>
</define>
<define name="xforms.mediatype">
  <element name="mediatype">
    <ref name="xforms.mediatype.attrib"/>
    <ref name="xforms.mediatype.content"/>
  </element>
</define>
<define name="xforms.mediatype.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Single.Node.Binding.attrib"/>
</define>
<define name="xforms.mediatype.content">
  <empty/>
</define>
<define name="xforms.output">
  <element name="output">
    <ref name="xforms.output.attrib"/>
    <ref name="xforms.output.content"/>
  </element>
</define>
<define name="xforms.output.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="UI.xforms.Common.attrib"/>
  <optional>
    <ref name="xforms.Single.Node.Binding.attrib"/>
  </optional>
  <optional>
    <attribute name="mediatype"/>
  </optional>
  <optional>
    <attribute name="value">
      <ref name="XPathExpression"/>
    </attribute>
  </optional>
</define>
<define name="xforms.output.content">
  <optional>
    <ref name="xforms.label"/>
  </optional>
</define>
<define name="xforms.input">
  <element name="input">
    <ref name="xforms.input.attrib"/>
    <ref name="xforms.input.content"/>
  </element>
</define>
<define name="xforms.input.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Single.Node.Binding.attrib"/>
  <optional>
    <attribute name="inputmode"/>
  </optional>
  <ref name="UI.xforms.Common.attrib"/>
```

```
  <optional>
    <attribute name="incremental">
      <data type="boolean"/>
    </attribute>
  </optional>
</define>
<define name="xforms.input.content">
  <ref name="xforms.label"/>
  <ref name="xforms.UI.Common.content"/>
</define>
<define name="xforms.textarea">
  <element name="textarea">
    <ref name="xforms.textarea.attrib"/>
    <ref name="xforms.textarea.content"/>
  </element>
</define>
<define name="xforms.textarea.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Single.Node.Binding.attrib"/>
  <optional>
    <attribute name="inputmode"/>
  </optional>
  <ref name="UI.xforms.Common.attrib"/>
  <optional>
    <attribute name="incremental">
      <data type="boolean"/>
    </attribute>
  </optional>
</define>
<define name="xforms.textarea.content">
  <ref name="xforms.label"/>
  <ref name="xforms.UI.Common.content"/>
</define>
<define name="xforms.secret">
  <element name="secret">
    <ref name="xforms.secret.attrib"/>
    <ref name="xforms.secret.content"/>
  </element>
</define>
<define name="xforms.secret.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Single.Node.Binding.attrib"/>
  <optional>
    <attribute name="inputmode"/>
  </optional>
  <ref name="UI.xforms.Common.attrib"/>
  <optional>
    <attribute name="incremental">
      <data type="boolean"/>
    </attribute>
  </optional>
</define>
<define name="xforms.secret.content">
  <ref name="xforms.label"/>
  <ref name="xforms.UI.Common.content"/>
</define>
<define name="xforms.upload">
  <element name="upload">
    <ref name="xforms.upload.attrib"/>
    <ref name="xforms.upload.content"/>
  </element>
</define>
<define name="xforms.upload.attrib" combine="interleave">
```

```
    <ref name="xforms.Common.attrib"/>
    <ref name="xforms.Single.Node.Binding.attrib"/>
    <optional>
      <attribute name="mediatype"/>
    </optional>
    <ref name="UI.xforms.Common.attrib"/>
    <optional>
      <attribute name="incremental">
        <data type="boolean"/>
      </attribute>
    </optional>
  </define>
  <define name="xforms.upload.content">
    <ref name="xforms.label"/>
    <optional>
      <ref name="xforms.filename"/>
    </optional>
    <optional>
      <ref name="xforms.mediatype"/>
    </optional>
    <ref name="xforms.UI.Common.content"/>
  </define>
  <define name="xforms.select1">
    <element name="select1">
      <ref name="xforms.select1.attrib"/>
      <ref name="xforms.select1.content"/>
    </element>
  </define>
  <define name="xforms.select1.attrib" combine="interleave">
    <ref name="xforms.Common.attrib"/>
    <ref name="xforms.Single.Node.Binding.attrib"/>
    <ref name="UI.xforms.Common.attrib"/>
    <optional>
      <attribute name="selection">
        <choice>
          <value>open</value>
          <value>closed</value>
        </choice>
      </attribute>
    </optional>
    <optional>
      <attribute name="incremental">
        <data type="boolean"/>
      </attribute>
    </optional>
  </define>
  <define name="xforms.select1.content">
    <ref name="xforms.label"/>
    <ref name="xforms.UI.Common.content"/>
    <ref name="xforms.List.UI.Common.content"/>
  </define>
  <define name="xforms.select">
    <element name="select">
      <ref name="xforms.select.attrib"/>
      <ref name="xforms.select.content"/>
    </element>
  </define>
  <define name="xforms.select.attrib" combine="interleave">
    <ref name="xforms.Common.attrib"/>
    <ref name="xforms.Single.Node.Binding.attrib"/>
    <ref name="UI.xforms.Common.attrib"/>
    <optional>
      <attribute name="selection">
```

```
      <choice>
        <value>open</value>
        <value>closed</value>
      </choice>
    </attribute>
  </optional>
  <optional>
    <attribute name="incremental">
      <data type="boolean"/>
    </attribute>
  </optional>
</define>
<define name="xforms.select.content">
  <ref name="xforms.label"/>
  <ref name="xforms.UI.Common.content"/>
  <ref name="xforms.List.UI.Common.content"/>
</define>
<define name="xforms.range">
  <element name="range">
    <ref name="xforms.range.attrib"/>
    <ref name="xforms.range.content"/>
  </element>
</define>
<define name="xforms.range.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.Single.Node.Binding.attrib"/>
  <ref name="UI.xforms.Common.attrib"/>
  <optional>
    <attribute name="start"/>
  </optional>
  <optional>
    <attribute name="end"/>
  </optional>
  <optional>
    <attribute name="step"/>
  </optional>
  <optional>
    <attribute name="incremental">
      <data type="boolean"/>
    </attribute>
  </optional>
</define>
<define name="xforms.range.content">
  <ref name="xforms.label"/>
  <ref name="xforms.UI.Common.content"/>
</define>
<define name="xforms.trigger">
  <element name="trigger">
    <ref name="xforms.trigger.attrib"/>
    <ref name="xforms.trigger.content"/>
  </element>
</define>
<define name="xforms.trigger.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <ref name="xforms.Single.Node.Binding.attrib"/>
  </optional>
  <optional>
    <ref name="UI.xforms.Common.attrib"/>
  </optional>
</define>
<define name="xforms.trigger.content">
  <ref name="xforms.label"/>
```

```
  <ref name="xforms.UI.Common.content"/>
</define>
<define name="xforms.submit">
  <element name="submit">
    <ref name="xforms.submit.attrib"/>
    <ref name="xforms.submit.content"/>
  </element>
</define>
<define name="xforms.submit.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <attribute name="submission">
    <data type="IDREF"/>
  </attribute>
  <optional>
    <ref name="xforms.Single.Node.Binding.attrib"/>
  </optional>
  <optional>
    <ref name="UI.xforms.Common.attrib"/>
  </optional>
</define>
<define name="xforms.submit.content">
  <ref name="xforms.label"/>
  <ref name="xforms.UI.Common.content"/>
</define>
<!-- Actions -->
<define name="xforms.action">
  <element name="action">
    <ref name="xforms.action.attrib"/>
    <ref name="xforms.action.content"/>
  </element>
</define>
<define name="xforms.action.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.action.content">
  <ref name="xforms.Actions"/>
</define>
<define name="xforms.dispatch">
  <element name="dispatch">
    <ref name="xforms.dispatch.attrib"/>
    <ref name="xforms.dispatch.content"/>
  </element>
</define>
<define name="xforms.dispatch.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <choice>
    <attribute name="name">
      <ref name="xforms.ActionName"/>
    </attribute>
    <element name="name">
      <ref name="xforms.ValueTemplate"/>
    </element>
  </choice>
  <choice>
    <!-- TODO collides with xhtml2-hypertext (xhtml.Common.attrib)
    <attribute name="target">
      <data type="IDREF"/>
    </attribute>
    -->
    <element name="target">
      <ref name="xforms.ValueTemplate"/>
    </element>
```

```
    </choice>
    <choice>
      <attribute name="delay">
        <data type="string"/>
      </attribute>
      <element name="delay">
        <ref name="xforms.ValueTemplate"/>
      </element>
    </choice>
    <optional>
      <attribute name="bubbles">
        <data type="boolean"/>
      </attribute>
    </optional>
    <optional>
      <attribute name="cancelable">
        <data type="boolean"/>
      </attribute>
    </optional>
    <ref name="xforms.actions.attrib"/>
  </define>
  <define name="xforms.dispatch.content">
    <empty/>
  </define>
  <define name="xforms.rebuild">
    <element name="rebuild">
      <ref name="xforms.rebuild.attrib"/>
      <ref name="xforms.rebuild.content"/>
    </element>
  </define>
  <define name="xforms.rebuild.attrib" combine="interleave">
    <ref name="xforms.Common.attrib"/>
    <optional>
      <attribute name="model">
        <data type="IDREF"/>
      </attribute>
    </optional>
    <ref name="xforms.actions.attrib"/>
  </define>
  <define name="xforms.rebuild.content">
    <empty/>
  </define>
  <define name="xforms.revalidate">
    <element name="revalidate">
      <ref name="xforms.revalidate.attrib"/>
      <ref name="xforms.revalidate.content"/>
    </element>
  </define>
  <define name="xforms.revalidate.attrib" combine="interleave">
    <ref name="xforms.Common.attrib"/>
    <optional>
      <attribute name="model">
        <data type="IDREF"/>
      </attribute>
    </optional>
    <ref name="xforms.actions.attrib"/>
  </define>
  <define name="xforms.revalidate.content">
    <empty/>
  </define>
  <define name="xforms.recalculate">
    <element name="recalculate">
      <ref name="xforms.recalculate.attrib"/>
```

```
    <ref name="xforms.recalculate.content"/>
  </element>
</define>
<define name="xforms.recalculate.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <attribute name="model">
      <data type="IDREF"/>
    </attribute>
  </optional>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.recalculate.content">
  <empty/>
</define>
<define name="xforms.refresh">
  <element name="refresh">
    <ref name="xforms.refresh.attrib"/>
    <ref name="xforms.refresh.content"/>
  </element>
</define>
<define name="xforms.refresh.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <attribute name="model">
      <data type="IDREF"/>
    </attribute>
  </optional>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.refresh.content">
  <empty/>
</define>
<define name="xforms.setfocus">
  <element name="setfocus">
    <ref name="xforms.setfocus.attrib"/>
    <ref name="xforms.setfocus.content"/>
  </element>
</define>
<define name="xforms.setfocus.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <choice>
    <attribute name="control">
      <data type="IDREF"/>
    </attribute>
    <element name="control">
      <ref name="xforms.ValueTemplate"/>
    </element>
  </choice>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.setfocus.content">
  <empty/>
</define>
<define name="xforms.load">
  <element name="load">
    <ref name="xforms.load.attrib"/>
    <ref name="xforms.load.content"/>
  </element>
</define>
<define name="xforms.load.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <choice>
```

```
    <!-- TODO collides with rdf/a resource attribute
    <attribute name="resource">
      <data type="anyURI"/>
    </attribute>
    -->
    <element name="resource">
      <ref name="xforms.ValueTemplate"/>
    </element>
  </choice>
  <optional>
    <attribute name="show">
      <choice>
        <value>new</value>
        <value>replace</value>
      </choice>
    </attribute>
  </optional>
  <ref name="xforms.Single.Node.Binding.attrib"/>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.load.content">
  <empty/>
</define>
<define name="xforms.setvalue">
  <element name="setvalue">
    <ref name="xforms.setvalue.attrib"/>
    <ref name="xforms.setvalue.content"/>
  </element>
</define>
<define name="xforms.setvalue.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <ref name="xforms.Single.Node.Binding.attrib"/>
  </optional>
  <optional>
    <attribute name="value">
      <ref name="XPathExpression"/>
    </attribute>
  </optional>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.setvalue.content">
  <text/>
</define>
<define name="xforms.send">
  <element name="send">
    <ref name="xforms.send.attrib"/>
    <ref name="xforms.send.content"/>
  </element>
</define>
<define name="xforms.send.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <attribute name="submission">
    <data type="IDREF"/>
  </attribute>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.send.content">
  <empty/>
</define>
<define name="xforms.reset">
  <element name="reset">
    <ref name="xforms.reset.attrib"/>
```

```
    <ref name="xforms.reset.content"/>
  </element>
</define>
<define name="xforms.reset.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <attribute name="model">
      <data type="IDREF"/>
    </attribute>
  </optional>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.reset.content">
  <empty/>
</define>
<define name="xforms.insert">
  <element name="insert">
    <ref name="xforms.insert.attrib"/>
    <ref name="xforms.insert.content"/>
  </element>
</define>
<define name="xforms.insert.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <attribute name="at">
    <ref name="XPathExpression"/>
  </attribute>
  <attribute name="position">
    <choice>
      <value>before</value>
      <value>after</value>
    </choice>
  </attribute>
  <optional>
    <ref name="xforms.Nodeset.Binding.attrib"/>
  </optional>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.insert.content">
  <empty/>
</define>
<define name="xforms.delete">
  <element name="delete">
    <ref name="xforms.delete.attrib"/>
    <ref name="xforms.delete.content"/>
  </element>
</define>
<define name="xforms.delete.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <attribute name="at">
    <ref name="XPathExpression"/>
  </attribute>
  <optional>
    <ref name="xforms.Nodeset.Binding.attrib"/>
  </optional>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.delete.content">
  <empty/>
</define>
<define name="xforms.setindex">
  <element name="setindex">
    <ref name="xforms.setindex.attrib"/>
    <ref name="xforms.setindex.content"/>
```

```
  </element>
</define>
<define name="xforms.setindex.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <attribute name="repeat">
    <data type="IDREF"/>
  </attribute>
  <attribute name="index">
    <ref name="XPathExpression"/>
  </attribute>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.setindex.content">
  <empty/>
</define>
<define name="xforms.toggle">
  <element name="toggle">
    <ref name="xforms.toggle.attrib"/>
    <ref name="xforms.toggle.content"/>
  </element>
</define>
<define name="xforms.toggle.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <choice>
    <attribute name="case">
      <data type="IDREF"/>
    </attribute>
    <element name="case">
      <ref name="xforms.ValueTemplate"/>
    </element>
  </choice>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.toggle.content">
  <empty/>
</define>
<define name="xforms.message">
  <element name="message">
    <ref name="xforms.message.attrib"/>
    <ref name="xforms.message.content"/>
  </element>
</define>
<define name="xforms.message.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <attribute name="level">
    <choice>
      <value>ephemeral</value>
      <value>modeless</value>
      <value>modal</value>
    </choice>
  </attribute>
  <optional>
    <ref name="xforms.Linking.attrib"/>
  </optional>
  <optional>
    <ref name="xforms.Single.Node.Binding.attrib"/>
  </optional>
  <ref name="xforms.actions.attrib"/>
</define>
<define name="xforms.message.content">
  <ref name="xforms.UI.Inline.content"/>
</define>
<define name="xforms.Actions">
```

```
    <zeroOrMore>
      <choice>
        <ref name="xforms.action"/>
        <ref name="xforms.dispatch"/>
        <ref name="xforms.rebuild"/>
        <ref name="xforms.recalculate"/>
        <ref name="xforms.refresh"/>
        <ref name="xforms.setfocus"/>
        <ref name="xforms.load"/>
        <ref name="xforms.setvalue"/>
        <ref name="xforms.send"/>
        <ref name="xforms.reset"/>
        <ref name="xforms.insert"/>
        <ref name="xforms.delete"/>
        <ref name="xforms.setindex"/>
        <ref name="xforms.toggle"/>
        <ref name="xforms.message"/>
      </choice>
    </zeroOrMore>
  </define>
  <!-- TODO: Do the XML Events attributes go only on actions? -->
  <define name="xforms.actions.attrib" combine="interleave">
    <ref name="xforms.Events.attrib"/>
    <optional>
      <attribute name="if">
        <ref name="xforms.ValueTemplate"/>
      </attribute>
    </optional>
    <optional>
      <attribute name="while">
        <ref name="xforms.ValueTemplate"/>
      </attribute>
    </optional>
  </define>
  <!-- Container Form Controls -->
  <define name="xforms.Container.Form.Controls">
    <choice>
      <ref name="xforms.group"/>
      <ref name="xforms.repeat"/>
      <ref name="xforms.switch"/>
    </choice>
  </define>
  <define name="xforms.repeat">
    <element name="repeat">
      <ref name="xforms.repeat.attrib"/>
      <ref name="xforms.repeat.content"/>
    </element>
  </define>
  <define name="xforms.repeat.attrib" combine="interleave">
    <ref name="xforms.Common.attrib"/>
    <ref name="xforms.Nodeset.Binding.attrib"/>
    <ref name="UI.xforms.Common.attrib"/>
    <optional>
      <attribute name="startindex">
        <data type="positiveInteger"/>
      </attribute>
    </optional>
    <optional>
      <attribute name="number">
        <data type="nonNegativeInteger"/>
      </attribute>
    </optional>
  </define>
```

```
<define name="xforms.repeat.content">
  <zeroOrMore>
    <choice>
      <ref name="xforms.Core.Form.Controls"/>
      <ref name="xforms.Container.Form.Controls"/>
      <ref name="xforms.UI.Inline.content"/>
    </choice>
  </zeroOrMore>
</define>
<define name="xforms.group">
  <element name="group">
    <ref name="xforms.group.attrib"/>
    <ref name="xforms.group.content"/>
  </element>
</define>
<define name="xforms.group.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <ref name="xforms.Single.Node.Binding.attrib"/>
  </optional>
  <ref name="UI.xforms.Common.attrib"/>
</define>
<define name="xforms.group.content">
  <optional>
    <ref name="xforms.label"/>
  </optional>
  <zeroOrMore>
    <choice>
      <ref name="xforms.Core.Form.Controls"/>
      <ref name="xforms.Container.Form.Controls"/>
      <ref name="xforms.UI.Common.content"/>
      <ref name="xforms.UI.Inline.content"/>
    </choice>
  </zeroOrMore>
</define>
<define name="xforms.switch">
  <element name="switch">
    <ref name="xforms.switch.attrib"/>
    <ref name="xforms.switch.content"/>
  </element>
</define>
<define name="xforms.switch.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
  <optional>
    <ref name="xforms.Single.Node.Binding.attrib"/>
  </optional>
  <ref name="UI.xforms.Common.attrib"/>
</define>
<define name="xforms.switch.content">
  <oneOrMore>
    <ref name="xforms.case"/>
  </oneOrMore>
</define>
<define name="xforms.case">
  <element name="case">
    <ref name="xforms.case.attrib"/>
    <ref name="xforms.case.content"/>
  </element>
</define>
<define name="xforms.case.attrib" combine="interleave">
  <ref name="xforms.Common.attrib"/>
</define>
<define name="xforms.case.content">
```

```
    <optional>
      <ref name="xforms.label"/>
    </optional>
    <zeroOrMore>
      <choice>
        <ref name="xforms.Core.Form.Controls"/>
        <ref name="xforms.Container.Form.Controls"/>
        <ref name="xforms.action"/>
        <ref name="xforms.UI.Inline.content"/>
      </choice>
    </zeroOrMore>
  </define>
  <!-- Simple Types -->
  <define name="QNameButNotNCName">
    <data type="QName">
      <param name="pattern">[^:]+:[^:]+</param>
    </data>
  </define>
  <define name="XPathExpression">
    <data type="string"/>
  </define>
  <define name="xforms.ValueTemplate">
    <ref name="XPathExpression"/>
  </define>
  <!-- Actions -->
  <define name="xforms.ActionName">
    <choice>
      <value>xforms-model-construct</value>
      <value>xforms-model-construct-done</value>
      <value>xforms-ready</value>
      <value>xforms-model-destruct</value>
      <value>xforms-rebuild</value>
      <value>xforms-recalculate</value>
      <value>xforms-revalidate</value>
      <value>xforms-refresh</value>
      <value>xforms-reset</value>
      <value>xforms-previous</value>
      <value>xforms-next</value>
      <value>xforms-focus</value>
      <value>xforms-help</value>
      <value>xforms-hint</value>
      <value>xforms-submit</value>
      <value>xforms-submit-serialize</value>
      <value>xforms-insert</value>
      <value>xforms-delete</value>
      <value>xforms-value-changed</value>
      <value>xforms-valid</value>
      <value>xforms-invalid</value>
      <value>xforms-readonly</value>
      <value>xforms-readwrite</value>
      <value>xforms-required</value>
      <value>xforms-optional</value>
      <value>xforms-enabled</value>
      <value>xforms-disabled</value>
      <value>xforms-select</value>
      <value>xforms-deselect</value>
      <value>xforms-in-range</value>
      <value>xforms-out-of-range</value>
      <value>xforms-scroll-first</value>
      <value>xforms-scroll-last</value>
      <value>xforms-submit-done</value>
      <value>xforms-binding-exception</value>
      <value>xforms-compute-exception</value>
```

```
        <value>xforms-link-error</value>
        <value>xforms-link-exception</value>
        <value>xforms-output-error</value>
        <value>xforms-submit-error</value>
        <value>xforms-version-exception</value>
        <data type="NMTOKEN">
          <except>
            <data type="NMTOKEN">
              <param name="pattern">^xforms-.*</param>
            </data>
          </except>
        </data>
      </choice>
    </define>
    <define name="anyElement">
      <element>
        <anyName/>
        <zeroOrMore>
          <choice>
            <attribute>
              <anyName/>
            </attribute>
            <text/>
            <ref name="anyElement"/>
          </choice>
        </zeroOrMore>
      </element>
    </define>
  </grammar>
```

C.4.8. XForms Repeat Attribute Collection

```
  <?xml version="1.0" encoding="UTF-8"?>
  <grammar xmlns="http://relaxng.org/ns/structure/1.0"
           xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"
           xmlns:x="http://www.w3.org/1999/xhtml">
    <x:h1>XForms Repeat Attribute Collection Module</x:h1>
    <define name="xforms.Repeat.attrib">
      <optional>
        <attribute name="repeat-model">
          <ref name="IDREF.datatype"/>
        </attribute>
      </optional>
      <optional>
        <attribute name="repeat-bind">
          <ref name="IDREF.datatype"/>
        </attribute>
      </optional>
      <optional>
        <attribute name="repeat-nodeset">
          <ref name="LocationPath.datatype"/>
        </attribute>
      </optional>
      <optional>
        <attribute name="repeat-startindex">
          <ref name="Number.datatype"/>
        </attribute>
      </optional>
      <optional>
```

```
        <attribute name="repeat-number">
          <ref name="Number.datatype"/>
        </attribute>
      </optional>
    </define>
  </grammar>
```

D. XHTML 2.0 Schema

This appendix is *normative*.

D.1. XHTML 2 Schema Driver

This section contains the driver for the XHTML 2 document type implementation as an XML Schema. Note that the structure and techniques in these driver files are as described in Modularization of XHTML [XHTMLMOD [p.311]].

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.w3.org/1999/xhtml"
    xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
    xmlns="http://www.w3.org/1999/xhtml"
    elementFormDefault="qualified" >
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema driver for XHTML 1.1.
      Please use this namespace for XHTML elements:
         "http://www.w3.org/1999/xhtml"
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
      This is XHTML, a reformulation of HTML as a modular XML application
      The Extensible HyperText Markup Language (XHTML)
      Copyright &#169;1998-2007 World Wide Web Consortium
      (Massachusetts Institute of Technology, European Research Consortium
       for Informatics and Mathematics, Keio University).
      All Rights Reserved.
      Permission to use, copy, modify and distribute the XHTML Schema
      modules and their accompanying xs: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
      these XML Schema modules for any purpose.
      They are provided "as is" without expressed or implied warranty.
    </xs:documentation>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
      This is the Schema Driver file for XHTML2
      Document Type
     This schema
        + imports external schemas (xml.xsd)
        + refedines (and include)s schema modules for XHTML2 Document Type.
        + includes Schema for Named content model for the
          XHTML2 Document Type
        XHTML2 Document Type includes the following Modules
           XHTML Core modules (Required for XHTML Family Conformance)
```

```
             +  text
             +  hypertext
             +  lists
             +  structure
            Other XHTML modules
             +  Edit
             +  Bdo
             +  Presentational
             +  Link
             +  Meta
             +  Base
             +  Scripting
             +  Style
             +  Image
             +  Applet
             +  Object
             +  Param (Applet/Object modules require Param Module)
             +  Tables
             +  Forms
             +  Client side image maps
             +  Server side image maps
             +  Ruby
    </xs:documentation>
    </xs:annotation>
    <xs:import
        namespace="http://www.w3.org/XML/1998/namespace"
        schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
         This import brings in the XML namespace attributes
         The XML attributes are used by various modules.
       </xs:documentation>
        </xs:annotation>
    </xs:import>
    <xs:import
        namespace="http://www.w3.org/1999/xhtml/datatypes/"
        schemaLocation="xhtml-datatypes-2.xsd"/>
    <xs:include
        schemaLocation="xhtml2-modules-1.xsd">
        <xs:annotation>
            <xs:documentation>
        Schema that includes all modules (and redefinitions)
        for XHTML2 Document Type.
           </xs:documentation>
        </xs:annotation>
    </xs:include>
    <xs:include
        schemaLocation="xhtml2-model-1.xsd">
        <xs:annotation>
            <xs:documentation>
        Document Model module for the XHTML2 Document Type.
        This schema file defines all named models used by XHTML
        Modularization Framework for XHTML2 Document Type
      </xs:documentation>
        </xs:annotation>
    </xs:include>
</xs:schema>
```

D.2. XHTML 2 Attribute Collections

This section contains a driver that accumulates all of the attribute collections and defines the Common collection based upon them.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
  <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
             schemaLocation="xhtml-datatypes-2.xsd" />
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
             schemaLocation="http://www.w3.org/2001/xml.xsd">
    <xs:annotation>
        <xs:documentation>
        This import brings in the XML namespace attributes
        The module itself does not provide the schemaLocation
        and expects the driver schema to provide the
        actual SchemaLocation.
        </xs:documentation>
    </xs:annotation>
  </xs:import>
  <xs:include schemaLocation="xhtml-core-2.xsd"/>
  <xs:include schemaLocation="xhtml-bidi-2.xsd"/>
  <xs:include schemaLocation="xml-events-attribs-2.xsd"/>
  <xs:include schemaLocation="xhtml-edit-2.xsd"/>
  <xs:include schemaLocation="xhtml-embedding-2.xsd"/>
  <!-- forms common collection -->
  <xs:include schemaLocation="xhtml-hyperAttributes-2.xsd"/>
  <xs:include schemaLocation="xhtml-i18n-2.xsd"/>
  <xs:include schemaLocation="xhtml-csismap-2.xsd"/>
  <xs:include schemaLocation="xhtml-media-attrib-2.xsd"/>
  <xs:include schemaLocation="xhtml-metaAttributes-2.xsd"/>
  <xs:include schemaLocation="xhtml-role-attrib-2.xsd"/>
  <xs:include schemaLocation="xhtml-style-attrib-2.xsd"/>
  <xs:attributeGroup name="xhtml.Core.extra.attrib"/>
  <xs:attributeGroup name="xhtml.Bidi.extra.attrib"/>
  <xs:attributeGroup name="xmlEvents.extra.attrib"/>
  <xs:attributeGroup name="xhtml.Edit.extra.attrib"/>
  <xs:attributeGroup name="xhtml.Embedding.extra.attrib"/>
  <xs:attributeGroup name="xhtml.Hypertext.extra.attrib"/>
  <xs:attributeGroup name="xhtml.I18n.extra.attrib"/>
  <xs:attributeGroup name="xhtml.Csismap.extra.attrib"/>
  <xs:attributeGroup name="xhtml.Media.extra.attrib"/>
  <xs:attributeGroup name="xhtml.MetaAttributes.extra.attrib"/>
  <xs:attributeGroup name="xhtml.Role.extra.attrib"/>
  <xs:attributeGroup name="xhtml.Style.extra.attrib"/>
  <xs:attributeGroup name="xhtml.CommonNoEvents.attrib">
      <xs:attributeGroup ref="xhtml.Core.attrib"/>
      <xs:attributeGroup ref="xhtml.Bidi.attrib"/>
      <xs:attributeGroup ref="xhtml.Edit.attrib"/>
      <xs:attributeGroup ref="xhtml.Embedding.attrib"/>
      <xs:attributeGroup ref="xhtml.Hypertext.attrib"/>
      <xs:attributeGroup ref="xhtml.I18n.attrib"/>
```

```
        <xs:attributeGroup ref="xhtml.ImageMap.attrib"/>
        <xs:attributeGroup ref="xhtml.Media.attrib"/>
        <xs:attributeGroup ref="xhtml.MetaAttributes.attrib"/>
        <xs:attributeGroup ref="xhtml.Role.attrib"/>
        <xs:attributeGroup ref="xhtml.Style.attrib"/>
        <xs:attributeGroup ref="xhtml.Common.extra"/>
    </xs:attributeGroup>
    <xs:attributeGroup name="xhtml.Common.attrib">
        <xs:attributeGroup ref="xhtml.CommonNoEvents.attrib"/>
        <xs:attributeGroup ref="xmlEvents.attrib"/>
    </xs:attributeGroup>
    <xs:attributeGroup name="xhtml.Global.Core.extra.attrib"/>
    <xs:attributeGroup name="xhtml.Global.Edit.extra.attrib"/>
    <xs:attributeGroup name="xhtml.Global.Embedding.extra.attrib"/>
    <xs:attributeGroup name="xhtml.Global.Hypertext.extra.attrib"/>
    <xs:attributeGroup name="xhtml.Global.Media.extra.attrib"/>
</xs:schema>
```

D.3. XHTML 2 Content Model

This section instantiates the XHTML 2 Content Model using XML Schema, including defining the various XHTML 2 Content Sets that are used by the modules.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
    elementFormDefault="qualified" >
    <xs:import
        namespace="http://www.w3.org/1999/xhtml/datatypes/"
        schemaLocation="xhtml-datatypes-2.xsd"/>
    <xs:annotation>
        <xs:documentation>
            This is the XML Schema module of common content models for XHTML11
            $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
        </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
            XHTML Document Model
            This module describes the groupings of elements/attributes
            that make up common content models for XHTML elements.
            XHTML has following basic content models:
               xhtml.Text.mix; character-level elements
               xhtml.Structural.mix; block-like elements, e.g., paragraphs and lists
               xhtml.Flow.mix; any block or inline elements
               xhtml.HeadOpts.mix; Head Elements
               xhtml.StructuralPre.mix; Special class for pre content model
            Any groups declared in this module may be used to create
            element content models, but the above are considered 'global'
            (insofar as that term applies here). XHTML has the
            following Attribute Groups
               xhtml.Core.extra.attrib
               xhtml.I18n.extra.attrib
```

```
            xhtml.Common.extra
        The above attribute Groups are considered Global
    </xs:documentation>
</xs:annotation>
<xs:attributeGroup
    name="xhtml.Common.extra">
    <xs:annotation>
        <xs:documentation> Extended Common Attributes </xs:documentation>
    </xs:annotation>
    </xs:attributeGroup>
<xs:attributeGroup
    name="xhtml.Global.Common.extra">
    <xs:annotation>
        <xs:documentation> Extended Global Common Attributes </xs:documentation>
    </xs:annotation>
</xs:attributeGroup>
<xs:group
    name="xhtml.Head.extra">
    <xs:sequence/>
</xs:group>
<xs:group
    name="xhtml.head.content">
    <xs:sequence>
        <xs:element
            name="title"
            minOccurs="1"
            maxOccurs="1"
            type="xhtml.title.type"/>
        <xs:choice>
            <xs:element
                name="access"
                type="xhtml.access.type"/>
            <xs:element
                name="action"
                type="xmlHandlers.action.type"/>
            <xs:element
                name="link"
                type="xhtml.link.type"/>
            <xs:element
                name="listener"
                type="xmlEvents.listener.type"/>
            <xs:element
                name="meta"
                type="xhtml.meta.type"/>
            <xs:element
                name="style"
                type="xhtml.style.type"/>
            <xs:group ref="xhtml.Head.extra"/>
        </xs:choice>
    </xs:sequence>
</xs:group>
<!-- Text Elements -->
<xs:group
    name="xhtml.Text.extra">
    <xs:sequence/>
</xs:group>
<xs:group
```

```
name="xhtml.TextNoRuby.class">
<xs:choice>
    <xs:element
        name="abbr"
        type="xhtml.abbr.type"/>
    <xs:element
        name="caption"
        type="xhtml.caption.type"/>
    <xs:element
        name="cite"
        type="xhtml.cite.type"/>
    <xs:element
        name="code"
        type="xhtml.code.type"/>
    <xs:element
        name="dfn"
        type="xhtml.dfn.type"/>
    <xs:element
        name="em"
        type="xhtml.em.type"/>
    <xs:element
        name="kbd"
        type="xhtml.kbd.type"/>
    <xs:element
        name="l"
        type="xhtml.l.type"/>
    <xs:element
        name="q"
        type="xhtml.q.type"/>
    <xs:element
        name="samp"
        type="xhtml.samp.type"/>
    <xs:element
        name="span"
        type="xhtml.span.type"/>
    <xs:element
        name="strong"
        type="xhtml.strong.type"/>
    <xs:element
        name="sub"
        type="xhtml.sub.type"/>
    <xs:element
        name="sup"
        type="xhtml.sup.type"/>
    <xs:element
        name="var"
        type="xhtml.var.type"/>
    <xs:element
        name="img"
        type="xhtml.img.type"/>
    <xs:element
        name="link"
        type="xhtml.link.type"/>
    <xs:element
        name="meta"
        type="xhtml.meta.type"/>
    <xs:element
```

```
            name="object"
            type="xhtml.object.type"/>
    </xs:choice>
</xs:group>
<xs:group name="xhtml.Text.class">
    <xs:choice>
        <xs:element
            name="ruby"
            type="xhtml.ruby.type"/>
        <xs:group
            ref="xhtml.TextNoRuby.class"/>
    </xs:choice>
</xs:group>
<xs:group name="xhtml.Text.mix">
    <xs:choice>
        <xs:group ref="xhtml.Text.class"/>
        <xs:group ref="xhtml.Text.extra"/>
    </xs:choice>
</xs:group>
<xs:group
    name="xhtml.Heading.class">
    <xs:choice>
        <xs:element
            name="h"
            type="xhtml.h.type"/>
        <xs:element
            name="h1"
            type="xhtml.h1.type"/>
        <xs:element
            name="h2"
            type="xhtml.h2.type"/>
        <xs:element
            name="h3"
            type="xhtml.h3.type"/>
        <xs:element
            name="h4"
            type="xhtml.h4.type"/>
        <xs:element
            name="h5"
            type="xhtml.h5.type"/>
        <xs:element
            name="h6"
            type="xhtml.h6.type"/>
    </xs:choice>
</xs:group>
<xs:group
    name="xhtml.List.class">
    <xs:choice>
        <xs:element
            name="ul"
            type="xhtml.ul.type"/>
        <xs:element
            name="ol"
            type="xhtml.ol.type"/>
        <xs:element
            name="dl"
            type="xhtml.dl.type"/>
```

```
    </xs:choice>
</xs:group>
<xs:group
    name="xhtml.List.extra">
    <xs:sequence/>
</xs:group>
<xs:group name="xhtml.List.mix">
    <xs:choice>
        <xs:group ref="xhtml.List.class"/>
        <xs:group ref="xhtml.List.extra"/>
    </xs:choice>
</xs:group>
<xs:group
    name="xhtml.Table.class">
    <xs:choice>
        <xs:element
            name="table"
            type="xhtml.table.type"/>
    </xs:choice>
</xs:group>
<xs:group
    name="xhtml.Structural.class">
    <xs:choice>
        <xs:element
            name="action"
            type="xmlHandlers.action.type"/>
        <xs:element
            name="address"
            type="xhtml.address.type"/>
        <xs:element
            name="blockcode"
            type="xhtml.blockcode.type"/>
        <xs:element
            name="blockquote"
            type="xhtml.blockquote.type"/>
        <xs:element
            name="div"
            type="xhtml.div.type"/>
        <xs:group ref="xhtml.List.class"/>
        <xs:element
            name="listener"
            type="xmlEvents.listener.type"/>
        <xs:element
            name="p"
            type="xhtml.p.type"/>
        <xs:element
            name="pre"
            type="xhtml.pre.type"/>
        <xs:element
            name="script"
            type="xmlScripting.script.type"/>
        <xs:element
            name="section"
            type="xhtml.section.type"/>
        <xs:element
            name="separator"
            type="xhtml.separator.type"/>
```

```
            <xs:element
                name="style"
                type="xhtml.style.type"/>
            <xs:element
                name="table"
                type="xhtml.table.type"/>
        </xs:choice>
    </xs:group>
    <xs:group
        name="xhtml.Structural.extra">
        <xs:sequence/>
    </xs:group>
    <!--
    Structural.mix includes all block elements plus %Misc.class;
  -->
    <xs:group name="xhtml.Structural.mix">
        <xs:choice>
            <xs:group ref="xhtml.Structural.class"/>
            <xs:group ref="xhtml.Structural.extra"/>
        </xs:choice>
    </xs:group>
    <xs:group
        name="xhtml.Flow.class">
        <xs:choice>
            <xs:group
                ref="xhtml.Heading.class"/>
            <xs:group
                ref="xhtml.Structural.class"/>
            <xs:group
                ref="xhtml.Text.class"/>
        </xs:choice>
    </xs:group>
    <!--
    All Content Elements
    Flow.mix includes all text content, block and inline
    Note that the "any" element included here allows us
    to add data from any other namespace, a necessity
    for compound document creation.
    Note however that it is not possible to add
    to any head level element without further
    modification. To add RDF metadata to the head
    of a document, modify the structure module.
  -->
    <xs:group name="xhtml.Flow.extra">
        <xs:sequence/>
    </xs:group>
    <xs:group
        name="xhtml.Flow.mix">
        <xs:choice>
            <xs:group
                ref="xhtml.Flow.class"/>
            <xs:group
                ref="xhtml.Flow.extra"/>
        </xs:choice>
    </xs:group>
```

```
    <xs:element
        name="html"
        type="xhtml.html.type"/>
</xs:schema>
```

D.4. XHTML 2 Modules

This section loads all of the modules that are used in XHTML 2. The modules themselves are defined in the next Appendix.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified"
           xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/" >
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
      schemaLocation="http://www.w3.org/2001/xml.xsd" />
    <xs:annotation>
        <xs:documentation>
      This schema includes all modules for XHTML1.1 Document Type.
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
     This schema includes all modules (and redefinitions)
     for XHTML2 Document Type.
     XHTML2 Document Type includes the following Modules
       XHTML Core modules (Required for XHTML Family Conformance)
            +  text
            +  hypertext
            +  lists
            +  structure
       Other XHTML modules
            +  Edit
            +  Bdo
            +  Presentational
            +  Link
            +  Meta
            +  Base
            +  Scripting
            +  Style
            +  Image
            +  Applet
            +  Object
            +  Param (Applet/Object modules require Param Module)
            +  Tables
            +  Target
            +  Forms
            +  Client side image maps
            +  Server side image maps
    </xs:documentation>
    </xs:annotation>
```

```
    <xs:include schemaLocation="xhtml2-attribs-1.xsd"/>
    <xs:include schemaLocation="xhtml-framework-2.xsd"/>
    <xs:include schemaLocation="xhtml-structural-2.xsd"/>
    <xs:include schemaLocation="xhtml-text-2.xsd"/>
    <xs:include schemaLocation="xhtml-hypertext-2.xsd"/>
    <xs:include schemaLocation="xhtml-list-2.xsd"/>
    <xs:redefine schemaLocation="xhtml-document-2.xsd">
        <xs:attributeGroup name="xhtml.version.attrib">
            <xs:attribute name="version" type="xh11d:CDATA" fixed="xhtml2"/>
        </xs:attributeGroup>
    </xs:redefine>
    <xs:include schemaLocation="xhtml-edit-2.xsd"/>
    <xs:include schemaLocation="xhtml-i18n-2.xsd"/>
    <xs:include schemaLocation="xhtml-meta-2.xsd"/>
    <xs:include schemaLocation="xhtml-legacy-br-2.xsd"/>
    <xs:include schemaLocation="xhtml-legacy-edit-2.xsd"/>
    <xs:include schemaLocation="xhtml-legacy-heading-2.xsd"/>
    <xs:redefine schemaLocation="xhtml-access-2.xsd">
        <xs:attributeGroup name="xhtmlAccess.Common.attrib">
            <xs:attributeGroup ref="xhtml.Common.attrib"/>
        </xs:attributeGroup>
    </xs:redefine>
    <xs:redefine schemaLocation="xml-events-2.xsd">
        <xs:attributeGroup name="xmlEvents.Common.attrib">
            <xs:attributeGroup ref="xhtml.CommonNoEvents.attrib"/>
        </xs:attributeGroup>
    </xs:redefine>
    <xs:redefine schemaLocation="xml-handlers-2.xsd">
        <xs:attributeGroup name="xmlHandlers.Common.attrib">
            <xs:attributeGroup ref="xhtml.Common.attrib"/>
        </xs:attributeGroup>
    </xs:redefine>
    <xs:redefine schemaLocation="xml-script-2.xsd">
        <xs:attributeGroup name="xmlScripting.Common.attrib"/>
        <xs:attributeGroup name="xmlScripting.script.attlist">
            <xs:attributeGroup ref="xhtml.Common.attrib"/>
            <xs:attribute name="charset" type="xh11d:Charset"/>
            <xs:attribute name="defer" type="xs:NMTOKEN" fixed="defer"/>
            <xs:attribute name="implements" type="xh11d:URIorSafeCURIEs"/>
        </xs:attributeGroup>
    </xs:redefine>
    <xs:include schemaLocation="xhtml-style-2.xsd"/>
    <xs:include schemaLocation="xhtml-image-2.xsd"/>
    <xs:include schemaLocation="xhtml-csismap-2.xsd"/>
    <xs:include schemaLocation="xhtml-object-2.xsd"/>
    <xs:include schemaLocation="xhtml-param-2.xsd"/>
    <xs:include schemaLocation="xhtml-table-2.xsd"/>
    <xs:include schemaLocation="xhtml-ruby-2.xsd"/>
    <xs:include schemaLocation="xml-events-2.xsd"/>
</xs:schema>
```

E. XHTML Schema Module Implementations

This appendix is *normative*.

This appendix contains implementations of the modules defined in this specification via XML Schema [XMLSCHEMA [p.311]].

E.1. Required Modules

E.1.1. Datatypes Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns="http://www.w3.org/1999/xhtml/datatypes/"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
 targetNamespace="http://www.w3.org/1999/xhtml/datatypes/"
 elementFormDefault="qualified"
>
    <xs:annotation>
        <xs:documentation>
          XHTML Datatypes
          This is the XML Schema datatypes module for XHTML
          Defines containers for the XHTML datatypes, many of
          these imported from other specifications and standards.
          $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
        </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstraction.html#s_common_attrtypes"/>
    </xs:annotation>
    <!-- nn for pixels or nn% for percentage length -->
    <xs:simpleType name="Length">
        <xs:union memberTypes="xs:nonNegativeInteger">
            <xs:simpleType>
                <xs:restriction base="xs:token">
                    <xs:pattern value="\d+[%]|\d*\.\d+[%]"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:union>
    </xs:simpleType>
    <!-- space-separated list of link types -->
    <xs:simpleType name="LinkTypes">
        <xs:list itemType="xs:NMTOKEN"/>
    </xs:simpleType>
    <!-- single or comma-separated list of media descriptors -->
    <xs:simpleType name="MediaDesc">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>
    <!-- pixel, percentage, or relative -->
    <xs:simpleType name="MultiLength">
        <xs:union memberTypes="xh11d:Length">
            <xs:simpleType>
                <xs:restriction base="xs:token">
                    <xs:pattern value="\d*\*"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:union>
    </xs:simpleType>
    <!-- one or more digits (NUMBER) -->
    <xs:simpleType name="Number">
        <xs:restriction base="xs:nonNegativeInteger"/>
    </xs:simpleType>
    <!-- integer representing length in pixels -->
    <xs:simpleType name="Pixels">
        <xs:restriction base="xs:nonNegativeInteger"/>
    </xs:simpleType>
    <!-- script expression -->
    <xs:simpleType name="Script">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>
    <!-- sixteen color names or RGB color expression-->
    <xs:simpleType name="Color">
        <xs:union memberTypes="xs:NMTOKEN">
            <xs:simpleType>
                <xs:restriction base="xs:token">
                    <xs:pattern value="#[0-9a-fA-F]{3}([0-9a-fA-F]{3})?"/>
                </xs:restriction>
            </xs:simpleType>
```

```
    </xs:union>
</xs:simpleType>
<!-- textual content -->
<xs:simpleType name="Text">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
<!-- Imported Datatypes  -->
<!-- a single character, as per section 2.2 of [XML] -->
<xs:simpleType name="Character">
    <xs:restriction base="xs:string">
        <xs:length value="1" fixed="true"/>
    </xs:restriction>
</xs:simpleType>
<!-- a character encoding, as per [RFC2045] -->
<xs:simpleType name="Charset">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
<!-- a space separated list of character encodings, as per [RFC2045] -->
<xs:simpleType name="Charsets">
    <xs:list itemType="Charset"/>
</xs:simpleType>
<!-- media type, as per [RFC2045] -->
<xs:simpleType name="ContentType">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
<!-- comma-separated list of media types, as per [RFC2045] -->
<xs:simpleType name="ContentTypes">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
<!-- date and time information. ISO date format -->
<xs:simpleType name="Datetime">
    <xs:restriction base="xs:dateTime"/>
</xs:simpleType>
<!-- formal public identifier, as per [ISO8879] -->
<xs:simpleType name="FPI">
    <xs:restriction base="xs:normalizedString"/>
</xs:simpleType>
<!-- a window name as used in the target attribute -->
<xs:simpleType name="FrameTarget">
  <xs:union>
    <xs:simpleType>
      <xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="_blank"/>
        <xs:enumeration value="_self"/>
        <xs:enumeration value="_parent"/>
        <xs:enumeration value="_top"/>
      </xs:restriction>
    </xs:simpleType>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="[a-zA-Z].*"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:union>
</xs:simpleType>
<!-- a language code, as per [RFC3066] -->
<xs:simpleType name="LanguageCode">
    <xs:restriction base="xs:language"/>
</xs:simpleType>
<!-- a comma separated list of language ranges -->
<xs:simpleType name="LanguageCodes">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
<!-- a Uniform Resource Identifier, see [URI] -->
<xs:simpleType name="URI">
    <xs:restriction base="xs:anyURI"/>
</xs:simpleType>
<!-- a space-separated list of Uniform Resource Identifiers, see [URI] -->
<xs:simpleType name="URIs">
    <xs:list itemType="xs:anyURI"/>
</xs:simpleType>
<!-- comma-separated list of MultiLength -->
<xs:simpleType name="MultiLengths">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
<!-- character Data -->
<xs:simpleType name="CDATA">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
<!-- CURIE placeholder datatypes -->
<xs:simpleType name="CURIE">
    <xs:restriction base="xs:string">
        <xs:pattern value="(([\i-[:]][\c-[:]]*)?:)?.+" />
        <xs:minLength value="1"/>
    </xs:restriction>
</xs:simpleType>
<xs:simpleType name="CURIEs">
    <xs:list itemType="xhtml:CURIE"/>
</xs:simpleType>
<xs:simpleType name="SafeCURIE">
```

```
    <xs:restriction base="xs:string">
        <xs:pattern value="\[(([\i-[:]][\c-[:]]*)?:)?.+\]" />
        <xs:minLength value="3"/>
    </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="SafeCURIEs">
        <xs:list itemType="xh11d:SafeCURIE"/>
    </xs:simpleType>
    <xs:simpleType name="URIorSafeCURIE">
        <xs:union memberTypes="xs:anyURI xh11d:SafeCURIE" />
    </xs:simpleType>
    <xs:simpleType name="URIorSafeCURIEs">
        <xs:list itemType="xh11d:URIorSafeCURIE"/>
    </xs:simpleType>
    <xs:simpleType name="Encodings">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>
    <xs:simpleType name="HrefTarget">
        <xs:restriction base="xs:NMTOKEN"/>
    </xs:simpleType>
    <xs:simpleType name="LocationPath">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>
    <xs:simpleType name="QName">
        <xs:restriction base="xs:string"/>
    </xs:simpleType>
    <xs:simpleType name="QNames">
        <xs:list itemType="xh11d:QName"/>
    </xs:simpleType>
</xs:schema>
```

E.1.2. Document Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/">
    <xs:import
        namespace="http://www.w3.org/1999/xhtml/datatypes/"
        schemaLocation="xhtml-datatypes-2.xsd"/>
    <xs:annotation>
        <xs:documentation>
        This is the XML Schema Document Module for XHTML
          * title, head, body, html
        The Document Module defines the major structural elements and
        their attributes.
        $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
      </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml2#s_documentmodule"/>
    </xs:annotation>
    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
        schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
                This import brings in the XML namespace attributes
                The module itself does not provide the schemaLocation
                and expects the driver schema to provide the
                actual SchemaLocation.
            </xs:documentation>
        </xs:annotation>
    </xs:import>
    <xs:attributeGroup
        name="xhtml.title.attlist">
        <xs:attributeGroup
            ref="xhtml.Common.attrib"/>
```

```
</xs:attributeGroup>
<xs:group
    name="xhtml.title.content">
    <xs:sequence/>
</xs:group>
<xs:complexType
    name="xhtml.title.type"
    mixed="true">
    <xs:group
        ref="xhtml.title.content"/>
    <xs:attributeGroup
        ref="xhtml.title.attlist"/>
</xs:complexType>
<xs:attributeGroup
    name="xhtml.profile.attrib">
    <xs:attribute
        name="profile"
        type="xh11d:URIs"/>
</xs:attributeGroup>
<xs:attributeGroup
    name="xhtml.head.attlist">
    <xs:attributeGroup
        ref="xhtml.profile.attrib"/>
    <xs:attributeGroup
        ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:complexType
    name="xhtml.head.type">
    <xs:group
        ref="xhtml.head.content"/>
    <xs:attributeGroup
        ref="xhtml.head.attlist"/>
</xs:complexType>
<xs:attributeGroup
    name="xhtml.body.attlist">
    <xs:attributeGroup
        ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group
    name="xhtml.body.content">
    <xs:sequence>
        <xs:group
            ref="xhtml.Structural.mix"
            minOccurs="0"
            maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType
    name="xhtml.body.type">
    <xs:group
        ref="xhtml.body.content"/>
    <xs:attributeGroup
        ref="xhtml.body.attlist"/>
</xs:complexType>
<xs:attributeGroup
    name="xhtml.version.attrib">
    <xs:attribute
```

```
            name="version"
            type="xh11d:CDATA"/>
    </xs:attributeGroup>
    <xs:attributeGroup
        name="xhtml.html.attlist">
        <xs:attributeGroup
            ref="xhtml.version.attrib"/>
        <xs:attributeGroup
            ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group
        name="xhtml.html.content">
        <xs:sequence>
            <xs:element
                name="head"
                type="xhtml.head.type"/>
            <xs:element
                name="body"
                type="xhtml.body.type"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType
        name="xhtml.html.type">
        <xs:group
            ref="xhtml.html.content"/>
        <xs:attributeGroup
            ref="xhtml.html.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.1.3. Structural Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
          This is the XML Schema Structural module for XHTML
          $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
       </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
           Structural
           This module declares the elements and their attributes used to
           support content structural markup.
           * address, blockcode, blockquote, div, h,
             p, pre, section, separator
      </xs:documentation>
      <xs:documentation source="http://www.w3.org/TR/xhtml2#s_structuralmodule"/>
```

```
  </xs:annotation>
<xs:import namespace="http://www.w3.org/XML/1998/namespace"
           schemaLocation="http://www.w3.org/2001/xml.xsd">
      <xs:annotation>
          <xs:documentation>
        This import brings in the XML namespace attributes
        The module itself does not provide the schemaLocation
        and expects the driver schema to provide the
        actual SchemaLocation.
      </xs:documentation>
      </xs:annotation>
  </xs:import>
  <!-- address -->
  <xs:attributeGroup name="xhtml.address.attlist">
      <xs:attributeGroup ref="xhtml.Common.attrib"/>
  </xs:attributeGroup>
  <xs:group name="xhtml.address.content">
      <xs:sequence>
          <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
  </xs:group>
  <xs:complexType name="xhtml.address.type" mixed="true">
      <xs:group ref="xhtml.address.content"/>
      <xs:attributeGroup ref="xhtml.address.attlist"/>
  </xs:complexType>
  <!-- blockquote -->
  <xs:attributeGroup name="xhtml.blockquote.attlist">
      <xs:attributeGroup ref="xhtml.Common.attrib"/>
  </xs:attributeGroup>
  <xs:group name="xhtml.blockquote.content">
      <xs:choice>
          <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
      </xs:choice>
  </xs:group>
  <xs:complexType name="xhtml.blockquote.type">
      <xs:group ref="xhtml.blockquote.content"/>
      <xs:attributeGroup ref="xhtml.blockquote.attlist"/>
  </xs:complexType>
  <!-- blockcode -->
  <xs:attributeGroup name="xhtml.blockcode.attlist">
      <xs:attributeGroup ref="xhtml.Common.attrib"/>
  </xs:attributeGroup>
  <xs:group name="xhtml.blockcode.content">
      <xs:choice>
          <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
      </xs:choice>
  </xs:group>
  <xs:complexType name="xhtml.blockcode.type">
      <xs:group ref="xhtml.blockcode.content"/>
      <xs:attributeGroup ref="xhtml.blockcode.attlist"/>
  </xs:complexType>
  <!-- div -->
  <xs:attributeGroup name="xhtml.div.attlist">
      <xs:attributeGroup ref="xhtml.Common.attrib"/>
  </xs:attributeGroup>
  <xs:group name="xhtml.div.content">
      <xs:choice>
```

```
        <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:choice>
</xs:group>
<xs:complexType name="xhtml.div.type">
    <xs:group ref="xhtml.div.content"/>
    <xs:attributeGroup ref="xhtml.div.attlist"/>
</xs:complexType>
<!-- h -->
<xs:attributeGroup name="xhtml.h.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.h.content">
    <xs:choice>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:choice>
</xs:group>
<xs:complexType name="xhtml.h.type">
    <xs:group ref="xhtml.h.content"/>
    <xs:attributeGroup ref="xhtml.h.attlist"/>
</xs:complexType>
<!-- Heading Elements  -->
<xs:attributeGroup name="xhtml.heading.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:complexType name="xhtml.heading.type" mixed="true">
    <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    <xs:attributeGroup ref="xhtml.heading.attlist"/>
</xs:complexType>
<!-- p -->
<xs:attributeGroup name="xhtml.p.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.p.content">
    <xs:choice>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
        <xs:group ref="xhtml.List.mix" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="blockcode" type="xhtml.blockcode.type"/>
        <xs:element name="blockquote" type="xhtml.blockquote.type"/>
        <xs:element name="pre" type="xhtml.pre.type"/>
        <xs:element name="table" type="xhtml.table.type"/>
    </xs:choice>
</xs:group>
<xs:complexType name="xhtml.p.type" mixed="true">
    <xs:group ref="xhtml.p.content"/>
    <xs:attributeGroup ref="xhtml.p.attlist"/>
</xs:complexType>
<!-- pre -->
<xs:attributeGroup name="xhtml.pre.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.pre.content">
    <xs:choice>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
        <xs:group ref="xhtml.List.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:choice>
</xs:group>
<xs:complexType name="xhtml.pre.type" mixed="true">
```

```
        <xs:group ref="xhtml.pre.content"/>
        <xs:attributeGroup ref="xhtml.pre.attlist"/>
    </xs:complexType>
    <!-- section -->
    <xs:attributeGroup name="xhtml.section.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.section.content">
        <xs:sequence>
            <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.section.type" mixed="true">
        <xs:group ref="xhtml.section.content"/>
        <xs:attributeGroup ref="xhtml.section.attlist"/>
    </xs:complexType>
    <!-- separator -->
    <xs:attributeGroup name="xhtml.separator.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.separator.content">
        <xs:sequence/>
    </xs:group>
    <xs:complexType name="xhtml.separator.type" mixed="true">
        <xs:group ref="xhtml.separator.content"/>
        <xs:attributeGroup ref="xhtml.separator.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.1.4. Text Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
         This is the XML Schema Text module for XHTML
         $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
      Text
      This module declares the elements and their attributes used to
      support textual markup.
      * abbr, cite, code, dfn, em, kbd, l, q, samp, span, strong,
        sub, sup, var
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
    <xs:documentation source="http://www.w3.org/TR/xhtml2#s_textmodule"/>
```

```
</xs:annotation>
<xs:attributeGroup name="xhtml.abbr.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
    <xs:attribute name="full" type="xs:IDREF"/>
</xs:attributeGroup>
<xs:group name="xhtml.abbr.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.abbr.type" mixed="true">
    <xs:group ref="xhtml.abbr.content"/>
    <xs:attributeGroup ref="xhtml.abbr.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.cite.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.cite.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.cite.type" mixed="true">
    <xs:group ref="xhtml.cite.content"/>
    <xs:attributeGroup ref="xhtml.cite.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.code.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.code.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.code.type" mixed="true">
    <xs:group ref="xhtml.code.content"/>
    <xs:attributeGroup ref="xhtml.code.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.dfn.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.dfn.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.dfn.type" mixed="true">
    <xs:group ref="xhtml.dfn.content"/>
    <xs:attributeGroup ref="xhtml.dfn.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.em.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.em.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
```

```
</xs:group>
<xs:complexType name="xhtml.em.type" mixed="true">
    <xs:group ref="xhtml.em.content"/>
    <xs:attributeGroup ref="xhtml.em.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.kbd.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.kbd.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.kbd.type" mixed="true">
    <xs:group ref="xhtml.kbd.content"/>
    <xs:attributeGroup ref="xhtml.kbd.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.l.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.l.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.l.type" mixed="true">
    <xs:group ref="xhtml.l.content"/>
    <xs:attributeGroup ref="xhtml.l.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.q.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.q.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.q.type" mixed="true">
    <xs:group ref="xhtml.q.content"/>
    <xs:attributeGroup ref="xhtml.q.attlist"/>
</xs:complexType>
<!-- samp -->
<xs:attributeGroup name="xhtml.samp.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.samp.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.samp.type" mixed="true">
    <xs:group ref="xhtml.samp.content"/>
    <xs:attributeGroup ref="xhtml.samp.attlist"/>
</xs:complexType>
<!-- span -->
<xs:attributeGroup name="xhtml.span.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
```

```
</xs:attributeGroup>
<xs:group name="xhtml.span.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.span.type" mixed="true">
    <xs:group ref="xhtml.span.content"/>
    <xs:attributeGroup ref="xhtml.span.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.strong.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.strong.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.strong.type" mixed="true">
    <xs:group ref="xhtml.strong.content"/>
    <xs:attributeGroup ref="xhtml.strong.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.sub.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.sub.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.sub.type" mixed="true">
    <xs:group ref="xhtml.sub.content"/>
    <xs:attributeGroup ref="xhtml.sub.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.sup.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.sup.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.sup.type" mixed="true">
    <xs:group ref="xhtml.sup.content"/>
    <xs:attributeGroup ref="xhtml.sup.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.var.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.var.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.var.type" mixed="true">
```

```
        <xs:group ref="xhtml.var.content"/>
        <xs:attributeGroup ref="xhtml.var.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.1.5. Hypertext Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      Hypertext Module
      This is the XML Schema Hypertext module for XHTML
        * a
      This module declares the anchor ('a') element type, which
      defines the source of a hypertext link. The destination
      (or link 'target') is identified via its 'id' attribute
      rather than the 'name' attribute as was used in HTML.
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_hypertextmodule"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.a.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.a.content">
        <xs:sequence>
            <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.a.type" mixed="true">
        <xs:group ref="xhtml.a.content"/>
        <xs:attributeGroup ref="xhtml.a.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.1.6. List Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      List Module
      This is the XML Schema Lists module for XHTML
      List Module Elements
        * dl, di, dt, dd, ol, ul, li
      This module declares the list-oriented element types
      and their attributes.
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml2#s_listmodule"/>
    </xs:annotation>
    <!-- dt -->
    <xs:attributeGroup name="xhtml.dt.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.dt.content">
        <xs:sequence>
            <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
```

```
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.dt.type" mixed="true">
    <xs:group ref="xhtml.dt.content"/>
    <xs:attributeGroup ref="xhtml.dt.attlist"/>
</xs:complexType>
<!-- dd -->
<xs:attributeGroup name="xhtml.dd.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.dd.content">
    <xs:sequence>
        <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.dd.type" mixed="true">
    <xs:group ref="xhtml.dd.content"/>
    <xs:attributeGroup ref="xhtml.dd.attlist"/>
</xs:complexType>
<!-- di -->
<xs:attributeGroup name="xhtml.di.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.di.content">
    <xs:sequence>
        <xs:element name="dt" type="xhtml.dt.type" minOccurs="1" maxOccurs="unbounded"/>
        <xs:element name="dd" type="xhtml.dd.type" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.di.type" mixed="true">
    <xs:group ref="xhtml.di.content"/>
    <xs:attributeGroup ref="xhtml.di.attlist"/>
</xs:complexType>
<!-- dl -->
<xs:attributeGroup name="xhtml.dl.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.dl.content">
    <xs:sequence>
        <xs:element name="title" type="xhtml.title.type" minOccurs="0" maxOccurs="1"/>
        <xs:element name="caption" type="xhtml.caption.type" minOccurs="0" maxOccurs="1"/>
        <xs:choice maxOccurs="unbounded">
            <xs:choice minOccurs="1" maxOccurs="unbounded">
                <xs:element name="dt" type="xhtml.dt.type"/>
                <xs:element name="dd" type="xhtml.dd.type"/>
            </xs:choice>
            <xs:element name="di" type="xhtml.di.type" minOccurs="1" maxOccurs="unbounded" />
        </xs:choice>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.dl.type">
    <xs:group ref="xhtml.dl.content"/>
    <xs:attributeGroup ref="xhtml.dl.attlist"/>
</xs:complexType>
<!-- li -->
<xs:attributeGroup name="xhtml.li.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.li.content">
    <xs:sequence>
        <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.li.type" mixed="true">
    <xs:group ref="xhtml.li.content"/>
    <xs:attributeGroup ref="xhtml.li.attlist"/>
```

```
    </xs:complexType>
    <!-- ol -->
    <xs:attributeGroup name="xhtml.ol.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.ol.content">
        <xs:sequence>
            <xs:element name="title" type="xhtml.title.type" minOccurs="0" maxOccurs="1"/>
            <xs:element name="caption" type="xhtml.caption.type" minOccurs="0" maxOccurs="1"/>
            <xs:element name="li" type="xhtml.li.type" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.ol.type">
        <xs:group ref="xhtml.ol.content"/>
        <xs:attributeGroup ref="xhtml.ol.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.ul.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.ul.content">
        <xs:sequence>
            <xs:element name="title" type="xhtml.title.type" minOccurs="0" maxOccurs="1"/>
            <xs:element name="caption" type="xhtml.caption.type" minOccurs="0" maxOccurs="1"/>
            <xs:element name="li" type="xhtml.li.type" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.ul.type">
        <xs:group ref="xhtml.ul.content"/>
        <xs:attributeGroup ref="xhtml.ul.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.1.7. Core Attributes Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema core attributes module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml2/mod-core.html"/>
    </xs:annotation>
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
             schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
        This import brings in the XML namespace attributes
        The module itself does not provide the schemaLocation
        and expects the driver schema to provide the
        actual SchemaLocation.
      </xs:documentation>
        </xs:annotation>
    </xs:import>
    <xs:attributeGroup name="xhtml.id">
```

```
        <xs:attribute name="id" type="xs:ID"/>
    </xs:attributeGroup>
    <xs:attributeGroup name="xhtml.class">
        <xs:attribute name="class" type="xs:NMTOKENS"/>
    </xs:attributeGroup>
    <xs:simpleType name="xhtml.layout.Datatype">
        <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="relevant"/>
            <xs:enumeration value="irrelevant"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:attributeGroup name="xhtml.layout">
        <xs:attribute name="layout" type="xhtml.layout.Datatype" default="irrelevant"/>
    </xs:attributeGroup>
    <xs:attributeGroup name="xhtml.title">
        <xs:attribute name="title" type="xs:string"/>
    </xs:attributeGroup>
    <xs:attributeGroup name="xhtml.Core.attrib">
        <!-- <xs:attribute ref="xml:space" fixed="preserve"/> -->
        <xs:attributeGroup ref="xhtml.id"/>
        <xs:attributeGroup ref="xhtml.class"/>
        <xs:attributeGroup ref="xhtml.layout"/>
        <xs:attributeGroup ref="xhtml.title"/>
        <xs:attributeGroup ref="xhtml.Core.extra.attrib"/>
    </xs:attributeGroup>
    <!-- Global attributes -->
    <xs:attribute name="id" type="xs:ID"/>
    <xs:attribute name="class" type="xs:NMTOKENS"/>
    <xs:attribute name="title" type="xs:string"/>
    <xs:attribute name="layout" type="xhtml.layout.Datatype"/>
    <xs:attributeGroup name="xhtml.Global.Core.attrib">
        <xs:attribute ref="id"/>
        <xs:attribute ref="class"/>
        <xs:attribute ref="layout"/>
        <xs:attribute ref="title"/>
        <xs:attributeGroup ref="xhtml.Global.Core.extra.attrib" />
    </xs:attributeGroup>
</xs:schema>
```

E.1.8. Hypertext Attributes Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      Hypertext Attributes Module
      This is the XML Schema Hypertext Attributes module for XHTML
      * cite, href, hreflang, hrefmedia, hreftype, nextfocus, prevfocus,
        and xml:base
        This module declares attributes and the Hypertext attribute
        collection.
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_hypertextmodule"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.Hypertext.attrib">
        <xs:attribute name="cite" type="xh11d:URI"/>
        <xs:attribute name="href" type="xh11d:URI"/>
        <xs:attribute name="hreflang" type="xh11d:LanguageCodes"/>
        <xs:attribute name="hrefmedia" type="xh11d:MediaDesc"/>
        <xs:attribute name="hreftype" type="xh11d:ContentTypes"/>
        <xs:attribute name="nextfocus" type="xs:IDREF"/>
```

```
        <xs:attribute name="prevfocus" type="xs:IDREF"/>
        <xs:attributeGroup ref="xhtml.Hypertext.extra.attrib" />
    </xs:attributeGroup>
    <!-- global attributes -->
    <xs:attribute name="cite" type="xh11d:URI"/>
    <xs:attribute name="href" type="xh11d:URI"/>
    <xs:attribute name="hreflang" type="xh11d:LanguageCodes"/>
    <xs:attribute name="hrefmedia" type="xh11d:MediaDesc"/>
    <xs:attribute name="hreftype" type="xh11d:ContentTypes"/>
    <xs:attribute name="nextfocus" type="xs:IDREF"/>
    <xs:attribute name="prevfocus" type="xs:IDREF"/>
    <xs:attributeGroup name="xhtml.Global.Hypertext.attrib">
        <xs:attribute ref="cite" />
        <xs:attribute ref="href" />
        <xs:attribute ref="hreflang" />
        <xs:attribute ref="hrefmedia" />
        <xs:attribute ref="hreftype" />
        <xs:attribute ref="nextfocus" />
        <xs:attribute ref="prevfocus" />
        <xs:attributeGroup ref="xhtml.Global.Hypertext.extra.attrib" />
    </xs:attributeGroup>
</xs:schema>
```

E.1.9. I18N Attribute Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema I18N attributes module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml2/mod-i18n.html"/>
    </xs:annotation>
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
             schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
        This import brings in the XML namespace attributes
        The module itself does not provide the schemaLocation
        and expects the driver schema to provide the
        actual SchemaLocation.
      </xs:documentation>
        </xs:annotation>
    </xs:import>
    <xs:attributeGroup name="xhtml.I18n.attrib">
        <xs:attribute ref="xml:lang" />
        <xs:attributeGroup ref="xhtml.I18n.extra.attrib"/>
    </xs:attributeGroup>
</xs:schema>
```

E.2. Optional Modules

E.2.1. Bi-directional Text Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema Bidirectional Text attribute module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml2/mod-bidi.html"/>
    </xs:annotation>
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
             schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
        This import brings in the XML namespace attributes
        The module itself does not provide the schemaLocation
        and expects the driver schema to provide the
        actual SchemaLocation.
      </xs:documentation>
        </xs:annotation>
  </xs:import>
    <xs:simpleType name="xhtml.Direction.datatype">
        <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="ltr"/>
            <xs:enumeration value="rtl"/>
            <xs:enumeration value="lro"/>
            <xs:enumeration value="rlo"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:attributeGroup name="xhtml.Bidi.attrib">
        <xs:attribute name="dir" default="ltr" type="xhtml.Direction.datatype"/>
    </xs:attributeGroup>
</xs:schema>
```

E.2.2. Caption Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
          This is the XML Schema Caption module for XHTML
          $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
        </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
```

```
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
           * caption
           This module declares the caption element.
        </xs:documentation>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_tablemodule"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.caption.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.caption.content">
        <xs:sequence>
            <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.caption.type" mixed="true">
        <xs:group ref="xhtml.caption.content"/>
        <xs:attributeGroup ref="xhtml.caption.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.2.3. Edit Attributes Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
          Editing Elements
          This is the XML Schema Editing Markup module for XHTML
           * edit, datetime
         This module declares attributes used to indicate
         inserted and deleted content while editing a document.
          $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
        </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_editmodule"/>
    </xs:annotation>
    <xs:simpleType name="xhtml.Edit.datatype">
        <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="inserted"/>
            <xs:enumeration value="deleted"/>
            <xs:enumeration value="changed"/>
            <xs:enumeration value="moved"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:attributeGroup name="xhtml.Edit.attrib">
        <xs:attribute name="edit" default="changed" type="xhtml.Edit.datatype"/>
        <xs:attribute name="datetime" type="xh11d:Datetime"/>
        <xs:attributeGroup ref="xhtml.Edit.extra.attrib"/>
    </xs:attributeGroup>
    <xs:attribute name="edit" default="changed" type="xhtml.Edit.datatype"/>
    <xs:attribute name="datetime" type="xh11d:Datetime"/>
    <xs:attributeGroup name="xhtml.Global.Edit.attrib">
        <xs:attribute ref="edit"/>
        <xs:attribute ref="datetime"/>
        <xs:attributeGroup ref="xhtml.Global.Edit.extra.attrib"/>
    </xs:attributeGroup>
</xs:schema>
```

E.2.4. Embedding Attributes Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
```

```
      Embedding Attributes Module
      This is the XML Schema Embedding Attributes module for XHTML
      * encoding, src, srctype
        This module declares attributes and the Embedding attribute
        collection.
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml2/mod-embedding.html"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.Embedding.attrib">
        <xs:attribute name="encoding" type="xh11d:Encodings"/>
        <xs:attribute name="src" type="xh11d:URI"/>
        <xs:attribute name="srctype" type="xh11d:ContentTypes"/>
        <xs:attributeGroup ref="xhtml.Embedding.extra.attrib" />
    </xs:attributeGroup>
    <!-- global attributes -->
    <xs:attribute name="encoding" type="xh11d:Encodings"/>
    <xs:attribute name="src" type="xh11d:URI"/>
    <xs:attribute name="srctype" type="xh11d:ContentTypes"/>
    <xs:attributeGroup name="xhtml.Global.Embedding.attrib">
        <xs:attribute ref="encoding"/>
        <xs:attribute ref="src"/>
        <xs:attribute ref="srctype"/>
        <xs:attributeGroup ref="xhtml.Global.Embedding.extra.attrib" />
    </xs:attributeGroup>
</xs:schema>
```

E.2.5. Image Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      Images
      This is the XML Schema Images module for XHTML
        * img
      This module provides markup to support basic image embedding.
      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
      the 'alt' attribute, and avoid server-side image maps.
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_imagemodule"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.img.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.img.content">
        <xs:sequence>
            <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.img.type">
        <xs:group ref="xhtml.img.content"/>
        <xs:attributeGroup ref="xhtml.img.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.2.6. Image Map Attributes Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      Client-side Image Maps
      This is the XML Schema Client-side Image Maps module for XHTML
        * area, map
      This module declares elements and attributes to support client-side
      image maps.
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_imapmodule"/>
    </xs:annotation>
    <xs:simpleType name="xhtml.Shape.Datatype">
        <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="rect"/>
            <xs:enumeration value="circle"/>
            <xs:enumeration value="poly"/>
            <xs:enumeration value="default"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="xhtml.Coords.Datatype">
        <xs:restriction base="xh11d:Text"/>
    </xs:simpleType>
    <xs:attributeGroup name="xhtml.ImageMap.attrib">
        <xs:attribute name="usemap" type="xh11d:URI"/>
        <xs:attribute name="ismap" type="xs:NMTOKEN" fixed="ismap"/>
        <xs:attribute name="shape" type="xhtml.Shape.Datatype" default="rect"/>
        <xs:attribute name="coords" type="xhtml.Coords.Datatype"/>
    </xs:attributeGroup>
</xs:schema>
```

E.2.7. Media Attribute Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema core attributes module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml2/mod-core.html"/>
    </xs:annotation>
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
             schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
        This import brings in the XML namespace attributes
        The module itself does not provide the schemaLocation
        and expects the driver schema to provide the
        actual SchemaLocation.
      </xs:documentation>
        </xs:annotation>
    </xs:import>
```

```
    <xs:attributeGroup name="xhtml.Media.attrib">
        <xs:attribute name="media" type="xh11d:MediaDesc"/>
    </xs:attributeGroup>
    <!-- Global attributes -->
    <xs:attribute name="media" type="xh11d:MediaDesc"/>
    <xs:attributeGroup name="xhtml.Global.Media.attrib">
        <xs:attribute ref="media"/>
        <xs:attributeGroup ref="xhtml.Global.Media.extra.attrib" />
    </xs:attributeGroup>
</xs:schema>
```

E.2.8. Metainformation Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema Metainformation module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
      Meta Information
        * meta
      This module declares the meta element type and its attributes,
      used to provide declarative document metainformation.
    </xs:documentation>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_metamodule"/>
    </xs:annotation>
    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
        schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
                This import brings in the XML namespace attributes
                The module itself does not provide the schemaLocation
                and expects the driver schema to provide the
                actual SchemaLocation.
            </xs:documentation>
        </xs:annotation>
    </xs:import>
    <xs:attributeGroup name="xhtml.meta.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.meta.content">
        <xs:choice>
            <xs:element name="meta" type="xhtml.meta.type"/>
        </xs:choice>
    </xs:group>
    <xs:complexType name="xhtml.meta.type">
        <xs:group ref="xhtml.meta.content"/>
        <xs:attributeGroup ref="xhtml.meta.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.link.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.link.content">
        <xs:sequence/>
    </xs:group>
    <xs:complexType name="xhtml.link.type">
        <xs:group ref="xhtml.link.content"/>
        <xs:attributeGroup ref="xhtml.link.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.2.9. Metainformation Attributes Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
    elementFormDefault="qualified"
>
  <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
             schemaLocation="xhtml-datatypes-2.xsd" />
  <xs:annotation>
    <xs:documentation>
      This is the XML Schema Metainformation Attributes module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
    <xs:documentation source="xhtml-rdfa-copyright-1.xsd"/>
  </xs:annotation>
  <xs:annotation>
    <xs:documentation>
      XHTML Metainformation Attributes
    </xs:documentation>
  </xs:annotation>
  <xs:attribute name="about" type="xh11d:URIorSafeCURIE"/>
  <xs:attribute name="content" type="xh11d:CDATA"/>
  <xs:attribute name="datatype" type="xh11d:CURIE"/>
  <xs:attribute name="typeof" type="xh11d:CURIEs"/>
  <xs:attribute name="property" type="xh11d:CURIEs"/>
  <xs:attribute name="rel" type="xh11d:CURIEs"/>
  <xs:attribute name="resource" type="xh11d:URIorSafeCURIE"/>
  <xs:attribute name="rev" type="xh11d:CURIEs"/>
  <xs:attributeGroup name="xhtml.MetaAttributes.attrib">
    <xs:attribute name="about"/>
    <xs:attribute name="content"/>
    <xs:attribute name="datatype"/>
    <xs:attribute name="typeof"/>
    <xs:attribute name="property"/>
    <xs:attribute name="rel"/>
    <xs:attribute name="resource"/>
    <xs:attribute name="rev"/>
  </xs:attributeGroup>
</xs:schema>
```

E.2.10. Object Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
        <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
        <xs:annotation>
                <xs:documentation>
      This is the XML Schema Object module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
                <xs:documentation source="xhtml-copyright-1.xsd"/>
        </xs:annotation>
        <xs:annotation>
                <xs:documentation>
      This module declares the object element type and its attributes,
      used to embed external objects as part of XHTML pages. In the
      document, place param elements prior to the object elements
```

```
      that require their content.
      Note that use of this module requires instantiation of the
      Param Element Module prior to this module.
      Elements defined here:
        * object (param)
    </xs:documentation>
                <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_objectmodule"/>
        </xs:annotation>
        <xs:include schemaLocation="xhtml-param-2.xsd">
                <xs:annotation>
                        <xs:documentation>
        Param module
        Elements defined here:
          * param
      </xs:documentation>
                </xs:annotation>
    </xs:include>
    <xs:include schemaLocation="xhtml-caption-2.xsd">
        <xs:annotation>
            <xs:documentation>
                Caption module - defines the caption element
            </xs:documentation>
        </xs:annotation>
    </xs:include>
    <!-- standby -->
    <xs:attributeGroup name="xhtml.standby.attlist">
                <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.standby.content">
        <xs:sequence>
            <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.standby.type" mixed="true">
        <xs:group ref="xhtml.standby.content"/>
        <xs:attributeGroup ref="xhtml.standby.attlist"/>
    </xs:complexType>
    <!-- object -->
        <xs:attributeGroup name="xhtml.object.attlist">
                <xs:attributeGroup ref="xhtml.Common.attrib"/>
                <xs:attribute name="declare">
                        <xs:simpleType>
                                <xs:restriction base="xs:NMTOKEN">
                                        <xs:enumeration value="declare"/>
                                </xs:restriction>
                        </xs:simpleType>
                </xs:attribute>
                <xs:attribute name="archive" type="xh11d:URIs"/>
                <xs:attribute name="content-length" type="xh11d:Number"/>
        </xs:attributeGroup>
        <xs:group name="xhtml.object.content">
                <xs:sequence>
            <xs:element name="caption" type="xhtml.caption.type"
                minOccurs="0" maxOccurs="1"/>
            <xs:element name="standby" type="xhtml.standby.type"
                minOccurs="0" maxOccurs="1"/>
                        <xs:choice minOccurs="0" maxOccurs="unbounded">
                                <xs:element name="param" type="xhtml.param.type"/>
                                <xs:group ref="xhtml.Flow.mix"/>
                        </xs:choice>
                </xs:sequence>
        </xs:group>
        <xs:complexType name="xhtml.object.type" mixed="true">
                <xs:group ref="xhtml.object.content"/>
                <xs:attributeGroup ref="xhtml.object.attlist"/>
        </xs:complexType>
</xs:schema>
```

E.2.10.1. Param Attribute

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
                schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema Param Element module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
      </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
      Parameters for Java Applets and Embedded Objects
        * param
      This module provides declarations for the param element,
      used to provide named property values for the applet
      and object elements.
```

```
    </xs:documentation>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_objectmodule"/>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_appletmodule"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.param.attlist">
        <xs:attributeGroup ref="xhtml.id"/>
        <xs:attribute name="name" type="xh11d:CDATA" use="required"/>
        <xs:attribute name="value" type="xh11d:CDATA"/>
        <xs:attribute name="valuetype" default="data">
            <xs:simpleType>
                <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="data"/>
                    <xs:enumeration value="ref"/>
                    <xs:enumeration value="object"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="type" type="xh11d:ContentType"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.param.content">
        <xs:sequence/>
    </xs:group>
    <xs:complexType name="xhtml.param.type">
        <xs:group ref="xhtml.param.content"/>
        <xs:attributeGroup ref="xhtml.param.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.2.11. Style Sheet Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema Stylesheets module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
      Stylesheets
        * style
      This module declares the style element type and its attributes,
      used to embed stylesheet information in the document head element.
    </xs:documentation>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_stylemodule"/>
    </xs:annotation>
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
             schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
          This import brings in the XML namespace attributes
          The module itself does not provide the schemaLocation
          and expects the driver schema to provide the
          actual SchemaLocation.
        </xs:documentation>
        </xs:annotation>
    </xs:import>
    <xs:attributeGroup name="xhtml.style.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
        <xs:attribute name="disabled" type="xs:NMTOKEN" fixed="disabled"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.style.content">
        <xs:sequence/>
    </xs:group>
    <xs:complexType name="xhtml.style.type" mixed="true">
        <xs:group ref="xhtml.style.content"/>
        <xs:attributeGroup ref="xhtml.style.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.2.12. Style Attribute Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      Inline Style module
      This is the XML Schema Inline Style module for XHTML
         * styloe attribute
      This module declares the 'style' attribute, used to support inline
      style markup.
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_styleattributemodule"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.Style.attrib">
        <xs:attribute name="style" type="xh11d:CDATA"/>
    </xs:attributeGroup>
</xs:schema>
```

E.2.13. Tables Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
          This is the XML Schema Tables module for XHTML
          $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
        </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
          Tables
           * table, caption, thead, tfoot, tbody, colgroup, col, tr, th, td
          This module declares element types and attributes used to provide
          table markup similar to HTML 4.0, including features that enable
          better accessibility for non-visual user agents.
        </xs:documentation>
        <xs:documentation source="http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410/abstract_modules.html#s_tablemodule"/>
    </xs:annotation>
    <xs:include schemaLocation="xhtml-caption-2.xsd"/>
    <xs:attributeGroup name="xhtml.scope.attrib">
        <xs:attribute name="scope">
            <xs:simpleType>
                <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="row"/>
                    <xs:enumeration value="col"/>
                    <xs:enumeration value="rowgroup"/>
                    <xs:enumeration value="colgroup"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
    </xs:attributeGroup>
    <xs:attributeGroup name="xhtml.td.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
        <xs:attribute name="abbr" type="xh11d:Text"/>
        <xs:attribute name="axis" type="xh11d:CDATA"/>
        <xs:attribute name="colspan" type="xh11d:Number" default="1"/>
        <xs:attribute name="headers" type="xs:IDREFS"/>
        <xs:attribute name="rowspan" type="xh11d:Number" default="1"/>
        <xs:attributeGroup ref="xhtml.scope.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.td.content">
        <xs:sequence>
            <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.td.type" mixed="true">
        <xs:group ref="xhtml.td.content"/>
        <xs:attributeGroup ref="xhtml.td.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.th.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
```

```
        <xs:attribute name="abbr" type="xh11d:Text"/>
        <xs:attribute name="axis" type="xh11d:CDATA"/>
        <xs:attribute name="colspan" type="xh11d:Number" default="1"/>
        <xs:attribute name="headers" type="xs:IDREFS"/>
        <xs:attribute name="rowspan" type="xh11d:Number" default="1"/>
        <xs:attributeGroup ref="xhtml.scope.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.th.content">
        <xs:sequence>
            <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.th.type" mixed="true">
        <xs:group ref="xhtml.th.content"/>
        <xs:attributeGroup ref="xhtml.th.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.tr.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
<!--        <xs:attributeGroup ref="xhtml.XFormsRepeat.attrib"/> -->
    </xs:attributeGroup>
    <xs:group name="xhtml.tr.content">
        <xs:sequence>
            <xs:choice maxOccurs="unbounded">
                <xs:element name="th" type="xhtml.th.type"/>
                <xs:element name="td" type="xhtml.td.type"/>
            </xs:choice>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.tr.type">
        <xs:group ref="xhtml.tr.content"/>
        <xs:attributeGroup ref="xhtml.tr.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.col.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
        <xs:attribute name="span" type="xh11d:Number" default="1"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.col.content">
        <xs:sequence/>
    </xs:group>
    <xs:complexType name="xhtml.col.type">
        <xs:group ref="xhtml.col.content"/>
        <xs:attributeGroup ref="xhtml.col.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.colgroup.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
        <xs:attribute name="span" type="xh11d:Number" default="1"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.colgroup.content">
        <xs:sequence>
            <xs:element name="col" type="xhtml.col.type" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.colgroup.type">
        <xs:group ref="xhtml.colgroup.content"/>
        <xs:attributeGroup ref="xhtml.colgroup.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.tbody.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.tbody.content">
        <xs:sequence>
            <xs:element name="tr" type="xhtml.tr.type" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.tbody.type">
        <xs:group ref="xhtml.tbody.content"/>
        <xs:attributeGroup ref="xhtml.tbody.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.tfoot.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.tfoot.content">
        <xs:sequence>
            <xs:element name="tr" type="xhtml.tr.type" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.tfoot.type">
        <xs:group ref="xhtml.tfoot.content"/>
        <xs:attributeGroup ref="xhtml.tfoot.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.thead.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.thead.content">
        <xs:sequence>
            <xs:element name="tr" type="xhtml.tr.type" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.thead.type">
        <xs:group ref="xhtml.thead.content"/>
```

```
        <xs:attributeGroup ref="xhtml.thead.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.summary.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.summary.content">
        <xs:sequence>
            <xs:group ref="xhtml.Flow.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.summary.type" mixed="true">
        <xs:group ref="xhtml.summary.content"/>
        <xs:attributeGroup ref="xhtml.summary.attlist"/>
    </xs:complexType>
    <xs:attributeGroup name="xhtml.table.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.table.content">
        <xs:sequence>
            <xs:element name="title" type="xhtml.title.type" minOccurs="0"/>
            <xs:element name="caption" type="xhtml.caption.type" minOccurs="0"/>
            <xs:element name="summary" type="xhtml.summary.type" minOccurs="0"/>
            <xs:choice>
                <xs:element name="col" type="xhtml.col.type" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="colgroup" type="xhtml.colgroup.type" minOccurs="0" maxOccurs="unbounded"/>
            </xs:choice>
            <xs:choice>
                <xs:sequence>
                    <xs:element name="thead" type="xhtml.thead.type" minOccurs="0"/>
                    <xs:element name="tfoot" type="xhtml.tfoot.type" minOccurs="0"/>
                    <xs:element name="tbody" type="xhtml.tbody.type" maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:choice>
                    <xs:element name="tr" type="xhtml.tr.type" maxOccurs="unbounded"/>
                </xs:choice>
            </xs:choice>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.table.type">
        <xs:group ref="xhtml.table.content"/>
        <xs:attributeGroup ref="xhtml.table.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.3. Legacy Module

E.3.1. Legacy Line Break Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-1.xsd" />
    <xs:annotation>
        <xs:documentation>
         This is the XML Schema Legacy line break module
         $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.br.attlist">
        <xs:attributeGroup ref="xhtml.Core.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.br.content">
        <xs:sequence/>
    </xs:group>
    <xs:complexType name="xhtml.br.type">
```

```
        <xs:group ref="xhtml.br.content"/>
        <xs:attributeGroup ref="xhtml.br.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.3.2. Legacy Editing Module

```
Module xhtml-legacy-edit-2.xsd not found!
```

E.3.3. Legacy Headings Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
          This is the XML Schema Structural module for XHTML
          $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
       </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
    </xs:annotation>
    <xs:annotation>
        <xs:documentation>
           Structural
           This module declares the legacy heading elements.
           * h1, h2, h3, h4, h5, h6
      </xs:documentation>
      <xs:documentation source="http://www.w3.org/TR/xhtml2#s_structuralmodule"/>
    </xs:annotation>
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
             schemaLocation="http://www.w3.org/2001/xml.xsd">
        <xs:annotation>
            <xs:documentation>
          This import brings in the XML namespace attributes
          The module itself does not provide the schemaLocation
          and expects the driver schema to provide the
          actual SchemaLocation.
        </xs:documentation>
        </xs:annotation>
    </xs:import>
    <!-- Heading Elements  -->
    <xs:attributeGroup name="xhtml.h1.attlist">
        <xs:attributeGroup ref="xhtml.Common.attrib"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.h1.content">
        <xs:sequence>
            <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:group>
    <xs:complexType name="xhtml.h1.type" mixed="true">
        <xs:group ref="xhtml.h1.content"/>
```

```
    <xs:attributeGroup ref="xhtml.h1.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.h2.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.h2.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.h2.type" mixed="true">
    <xs:group ref="xhtml.h2.content"/>
    <xs:attributeGroup ref="xhtml.h2.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.h3.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.h3.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.h3.type" mixed="true">
    <xs:group ref="xhtml.h3.content"/>
    <xs:attributeGroup ref="xhtml.h3.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.h4.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.h4.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.h4.type" mixed="true">
    <xs:group ref="xhtml.h4.content"/>
    <xs:attributeGroup ref="xhtml.h4.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.h5.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.h5.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.h5.type" mixed="true">
    <xs:group ref="xhtml.h5.content"/>
    <xs:attributeGroup ref="xhtml.h5.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xhtml.h6.attlist">
    <xs:attributeGroup ref="xhtml.Common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.h6.content">
    <xs:sequence>
        <xs:group ref="xhtml.Text.mix" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
```

```
    </xs:group>
    <xs:complexType name="xhtml.h6.type" mixed="true">
        <xs:group ref="xhtml.h6.content"/>
        <xs:attributeGroup ref="xhtml.h6.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.4. Modules from Other Specifications

E.4.1. Access Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema module for XHTML Access
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml-role#A_role"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtmlAccess.Common.attrib">
        <xs:attribute name="id" type="xs:ID"/>
        <xs:attribute name="media" type="xh11d:MediaDesc"/>
        <xs:anyAttribute/>
    </xs:attributeGroup>
    <xs:attributeGroup name="xhtml.access.attlist">
        <xs:attribute name="activate" default="no">
            <xs:simpleType>
                <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="yes"/>
                    <xs:enumeration value="no"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="key" type="xh11d:Character"/>
        <xs:attribute name="order" default="document">
            <xs:simpleType>
                <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="document"/>
                    <xs:enumeration value="list"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="targetid">
            <xs:simpleType>
                <xs:list itemType="xs:IDREF"/>
            </xs:simpleType>
        </xs:attribute>
```

```
        <xs:attribute name="targetrole" type="xh11d:CURIEs"/>
    </xs:attributeGroup>
    <xs:group name="xhtml.access.content">
        <xs:sequence/>
    </xs:group>
    <xs:complexType name="xhtml.access.type">
        <xs:group ref="xhtml.access.content"/>
        <xs:attributeGroup ref="xhtml.access.attlist"/>
    </xs:complexType>
</xs:schema>
```

E.4.2. Role Attribute Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
    <xs:annotation>
        <xs:documentation>
      This is the XML Schema attribute module for XHTML Role
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
        <xs:documentation source="xhtml-copyright-1.xsd"/>
        <xs:documentation source="http://www.w3.org/TR/xhtml-role#A_role"/>
    </xs:annotation>
    <xs:attributeGroup name="xhtml.Role.attrib">
        <xs:attribute name="role" type="xh11d:CURIEs"/>
    </xs:attributeGroup>
</xs:schema>
```

E.4.3. Ruby Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
>
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
               schemaLocation="xhtml-datatypes-2.xsd" />
  <xs:annotation>
    <xs:documentation>
      This is the Ruby module for XHTML
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
    </xs:documentation>
    <xs:documentation source="xhtml-copyright-1.xsd"/>
  </xs:annotation>
  <xs:annotation>
    <xs:documentation>
      "Ruby" are short runs of text alongside the base text, typically
      used in East Asian documents to indicate pronunciation or to
      provide a short annotation. The full specification for Ruby is here:
```

```
      http://www.w3.org/TR/2001/REC-ruby-20010531/
    This module defines "Ruby " or "complex Ruby" as described
    in the specification:
      http://www.w3.org/TR/2001/REC-ruby-20010531/#complex
    Simple or Basic Ruby are defined in a separate module.
    This module declares the elements and their attributes used to
    support complex ruby annotation markup. Elements defined here
          * ruby, rbc, rtc, rb, rt, rp
    This module expects the document model to define the
    following content models
      + xhtml.TextNoRuby.class
  </xs:documentation>
  <xs:documentation
       source="http://www.w3.org/TR/2001/REC-ruby-20010531/"/>
</xs:annotation>
<xs:group name="xhtml.ruby.content.simple">
    <xs:sequence>
        <xs:element name="rb" type="xhtml.rb.type"/>
        <xs:choice>
            <xs:element name="rt" type="xhtml.rt.type"/>
            <xs:sequence>
                <xs:element name="rp" type="xhtml.rp.type"/>
                <xs:element name="rt" type="xhtml.rt.type"/>
                <xs:element name="rp" type="xhtml.rp.type"/>
            </xs:sequence>
        </xs:choice>
    </xs:sequence>
</xs:group>
<xs:group name="xhtml.ruby.content.complex">
  <xs:sequence>
    <xs:element name="rbc" type="xhtml.rbc.type"/>
    <xs:element name="rtc" type="xhtml.rtc.type" maxOccurs="2"/>
  </xs:sequence>
</xs:group>
<!--
 add to this group any common attributes for all Ruby elements
-->
<xs:attributeGroup name="xhtml.ruby.common.attrib"/>
<xs:group name="xhtml.ruby.content">
  <xs:choice>
    <xs:group ref="xhtml.ruby.content.simple"/>
    <xs:group ref="xhtml.ruby.content.complex"/>
  </xs:choice>
</xs:group>
<xs:complexType name="xhtml.ruby.type">
  <xs:group ref="xhtml.ruby.content"/>
  <xs:attributeGroup ref="xhtml.ruby.common.attrib"/>
</xs:complexType>
<!--
 rbc (ruby base component) element
-->
<xs:attributeGroup name="xhtml.rbc.attlist">
  <xs:attributeGroup ref="xhtml.ruby.common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.rbc.content">
  <xs:sequence>
    <xs:element name="rb" type="xhtml.rb.type" />
```

```
  </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.rbc.type">
  <xs:group ref="xhtml.rbc.content"/>
  <xs:attributeGroup ref="xhtml.rbc.attlist"/>
</xs:complexType>
<!--
 rtc (ruby text component) element
-->
<xs:attributeGroup name="xhtml.rtc.attlist">
  <xs:attributeGroup ref="xhtml.ruby.common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.rtc.content">
  <xs:sequence>
    <xs:element name="rt" type="xhtml.rt.type" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.rtc.type">
  <xs:group ref="xhtml.rt.content"/>
  <xs:attributeGroup ref="xhtml.rtc.attlist"/>
</xs:complexType>
<!--
 rb (ruby base) element
-->
<xs:attributeGroup name="xhtml.rb.attlist">
  <xs:attributeGroup ref="xhtml.ruby.common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.rb.content">
  <xs:sequence>
     <xs:group ref="xhtml.TextNoRuby.class" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.rb.type" mixed="true">
  <xs:group ref="xhtml.rb.content"/>
  <xs:attributeGroup ref="xhtml.rb.attlist"/>
</xs:complexType>
<!--
 rt (ruby text) element
-->
<xs:attributeGroup name="xhtml.rt.attlist">
  <xs:attributeGroup ref="xhtml.ruby.common.attrib"/>
  <xs:attribute name="rbspan" type="xh11d:Number" default="1"/>
</xs:attributeGroup>
<xs:group name="xhtml.rt.content">
  <xs:sequence>
     <xs:group ref="xhtml.TextNoRuby.class" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:group>
<xs:complexType name="xhtml.rt.type" mixed="true">
  <xs:group ref="xhtml.rt.content"/>
  <xs:attributeGroup ref="xhtml.rt.attlist"/>
</xs:complexType>
<!-- rp (ruby parenthesis) element  -->
<xs:attributeGroup name="xhtml.rp.attlist">
  <xs:attributeGroup ref="xhtml.ruby.common.attrib"/>
</xs:attributeGroup>
<xs:group name="xhtml.rp.content">
```

```
    <xs:sequence/>
  </xs:group>
  <xs:complexType name="xhtml.rp.type" mixed="true">
    <xs:group ref="xhtml.rp.content"/>
    <xs:attributeGroup ref="xhtml.rp.attlist"/>
  </xs:complexType>
</xs:schema>
```

E.4.4. XForms Modules

```
Module xforms-2.xsd not found!
```

E.4.5. XML Events Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
    xsi:schemaLocation="http://www.w3.org/2001/XMLSchema
                        http://www.w3.org/2001/XMLSchema.xsd"
    elementFormDefault="unqualified"
    blockDefault="#all"
    finalDefault="#all"
    attributeFormDefault="unqualified">
  <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
    schemaLocation="xhtml-datatypes-2.xsd" />
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
      schemaLocation="http://www.w3.org/2001/xml.xsd"/>
  <xs:annotation>
    <xs:documentation>
      This is the XML Schema for XML Events
      URI: http://www.w3.org/MarkUp/SCHEMA/xml-events-2.xsd
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
      Note that this module requires the definition of
      xmlEvents.Common.attrib to be a collection of attributes to
      add to each element.  This collection MUST contain an 'id'
      attribute.
    </xs:documentation>
    <xs:documentation source="xml-events-copyright-2.xsd"/>
  </xs:annotation>
  <xs:annotation>
    <xs:documentation>
      XML Events element listener
        This module defines the listener element for XML Events.
        This element can be used to define event listeners. This
        module relies upon the XmlEvents.attlist attribute group
        defined in xml-events-attribs-2.xsd.
    </xs:documentation>
  </xs:annotation>
  <xs:attributeGroup name="xmlEvents.Common.attrib">
    <xs:attribute name="id" type="xs:ID"/>
    <xs:anyAttribute/>
  </xs:attributeGroup>
  <xs:attributeGroup name="xmlEvents.listener.attlist">
    <xs:attributeGroup ref="xmlEvents.attrib" />
```

```
    <xs:attributeGroup ref="xmlEvents.Common.attrib"/>
  </xs:attributeGroup>
  <xs:complexType name="xmlEvents.listener.type">
    <xs:attributeGroup ref="xmlEvents.listener.attlist"/>
  </xs:complexType>
  <xs:element name="listener" type="xmlEvents.listener.type"/>
</xs:schema>
```

E.4.6. XML Handlers Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2001/XMLSchema
                        http://www.w3.org/2001/XMLSchema.xsd"
    elementFormDefault="unqualified"
    blockDefault="#all"
    finalDefault="#all"
    attributeFormDefault="unqualified">
  <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
    schemaLocation="xhtml-datatypes-2.xsd" />
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
      schemaLocation="http://www.w3.org/2001/xml.xsd"/>
  <xs:annotation>
    <xs:documentation>
      This is the XML Schema for XML Handlers
      URI: http://www.w3.org/MarkUp/SCHEMA/xml-handlers-1.xsd
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
      Note that this module requires the definition of
      xmlHandlers.Common.attrib to be a collection of attributes to
      add to each element.  This collection MUST contain an 'id'
      attribute.
    </xs:documentation>
    <xs:documentation source="xml-events-copyright-2.xsd"/>
  </xs:annotation>
  <xs:attributeGroup name="xmlHandlers.Common.attrib">
      <xs:attribute name="id" type="xs:ID"/>
      <xs:anyAttribute/>
  </xs:attributeGroup>
  <xs:attributeGroup name="xmlHandlers.action.attlist">
    <xs:attributeGroup ref="xmlHandlers.Common.attrib"/>
    <xs:attribute name="event" use="required" type="xs:QName"/>
    <xs:attribute name="eventTarget">
        <xs:simpleType>
            <xs:list itemType="xs:IDREF"/>
        </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="declare">
        <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
                <xs:enumeration value="declare"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="if" type="xs:normalizedString"/>
    <xs:attribute name="while" type="xs:normalizedString"/>
    <xs:anyAttribute />
  </xs:attributeGroup>
  <xs:group name="xmlHandlers.action.content">
```

```
    <xs:choice>
      <xs:element name="action" type="xmlHandlers.action.type"/>
      <xs:element name="dispatchEvent" type="xmlHandlers.dispatchEvent.type"/>
      <xs:element name="addEventListener" type="xmlHandlers.addEventListener.type"/>
      <xs:element name="removeEventListener" type="xmlHandlers.removeEventListener.type"/>
      <xs:element name="stopPropagation" type="xmlHandlers.stopPropagation.type"/>
      <xs:element name="preventDefault" type="xmlHandlers.preventDefault.type"/>
    </xs:choice>
</xs:group>
<xs:complexType name="xmlHandlers.action.type">
  <xs:group ref="xmlHandlers.action.content"/>
  <xs:attributeGroup ref="xmlHandlers.action.attlist"/>
</xs:complexType>
<xs:attributeGroup name="xmlHandlers.dispatchEvent.attlist">
  <xs:attributeGroup ref="xmlHandlers.Common.attrib"/>
  <xs:attribute name="eventType" type="xs:QName"/>
  <xs:attribute name="eventTarget">
      <xs:simpleType>
          <xs:list itemType="xs:IDREF"/>
      </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="bubbles">
      <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="bubbles"/>
          </xs:restriction>
      </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="cancelable">
      <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="cancelable"/>
          </xs:restriction>
      </xs:simpleType>
  </xs:attribute>
  <xs:anyAttribute />
</xs:attributeGroup>
<xs:complexType name="xmlHandlers.dispatchEvent.type">
  <xs:attributeGroup ref="xmlHandlers.dispatchEvent.attlist"/>
</xs:complexType>
<xs:element name="dispatchEvent" type="xmlHandlers.dispatchEvent.type"/>
<xs:attributeGroup name="xmlHandlers.addEventListener.attlist">
  <xs:attributeGroup ref="xmlHandlers.Common.attrib"/>
  <xs:attribute name="event" use="required" type="xs:QName"/>
  <xs:attribute name="handler" use="required" type="xs:IDREF"/>
  <xs:attribute name="phase" default="default">
      <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
              <xs:enumeration value="bubble"/>
              <xs:enumeration value="capture"/>
              <xs:enumeration value="default"/>
              <xs:enumeration value="target"/>
          </xs:restriction>
      </xs:simpleType>
  </xs:attribute>
  <xs:anyAttribute />
</xs:attributeGroup>
<xs:complexType name="xmlHandlers.addEventListener.type">
  <xs:attributeGroup ref="xmlHandlers.addEventListener.attlist"/>
</xs:complexType>
<xs:element name="addEventListener" type="xmlHandlers.addEventListener.type"/>
<xs:attributeGroup name="xmlHandlers.removeEventListener.attlist">
  <xs:attributeGroup ref="xmlHandlers.Common.attrib"/>
```

```
    <xs:attribute name="event" use="required" type="xs:QName"/>
    <xs:attribute name="handler" use="required" type="xs:IDREF"/>
    <xs:attribute name="phase" default="default">
        <xs:simpleType>
            <xs:restriction base="xs:NMTOKEN">
                <xs:enumeration value="bubble"/>
                <xs:enumeration value="capture"/>
                <xs:enumeration value="default"/>
                <xs:enumeration value="target"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
    <xs:anyAttribute />
  </xs:attributeGroup>
  <xs:complexType name="xmlHandlers.removeEventListener.type">
    <xs:attributeGroup ref="xmlHandlers.removeEventListener.attlist"/>
  </xs:complexType>
  <xs:element name="removeEventListener" type="xmlHandlers.removeEventListener.type"/>
  <xs:attributeGroup name="xmlHandlers.stopPropagation.attlist">
    <xs:attributeGroup ref="xmlHandlers.Common.attrib"/>
    <xs:attribute name="event" use="required" type="xs:QName"/>
    <xs:anyAttribute />
  </xs:attributeGroup>
  <xs:complexType name="xmlHandlers.stopPropagation.type">
    <xs:attributeGroup ref="xmlHandlers.stopPropagation.attlist"/>
  </xs:complexType>
  <xs:element name="stopPropagation" type="xmlHandlers.stopPropagation.type"/>
  <xs:attributeGroup name="xmlHandlers.preventDefault.attlist">
    <xs:attributeGroup ref="xmlHandlers.Common.attrib"/>
    <xs:attribute name="event" use="required" type="xs:QName"/>
    <xs:anyAttribute />
  </xs:attributeGroup>
  <xs:complexType name="xmlHandlers.preventDefault.type">
    <xs:attributeGroup ref="xmlHandlers.preventDefault.attlist"/>
  </xs:complexType>
  <xs:element name="preventDefault" type="xmlHandlers.preventDefault.type"/>
</xs:schema>
```

E.4.7. XML Scripting Module

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2001/XMLSchema
                        http://www.w3.org/2001/XMLSchema.xsd"
    elementFormDefault="unqualified"
    blockDefault="#all"
    finalDefault="#all"
    attributeFormDefault="unqualified">
  <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
             schemaLocation="xhtml-datatypes-2.xsd" />
  <xs:import namespace="http://www.w3.org/XML/1998/namespace"
      schemaLocation="http://www.w3.org/2001/xml.xsd" />
  <xs:annotation>
    <xs:documentation>
      This is the XML Schema for XML Scripting
      URI: http://www.w3.org/MarkUp/SCHEMA/xml-script-1.xsd
      $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
```

```
        Note that this module requires the definition of
        xmlScripting.Common.attrib to be a collection of attributes to
        add to each element.  This collection MUST contain an 'id'
        attribute.
      </xs:documentation>
      <xs:documentation source="xml-events-copyright-2.xsd"/>
    </xs:annotation>
    <xs:attributeGroup name="xmlScripting.Common.attrib">
      <xs:attribute name="id" type="xs:ID"/>
      <xs:anyAttribute/>
    </xs:attributeGroup>
    <xs:attributeGroup name="xmlScripting.script.attlist">
      <xs:attributeGroup ref="xmlScripting.Common.attrib"/>
      <xs:attribute name="charset" type="xh11d:Charset"/>
      <xs:attribute name="defer" type="xs:NMTOKEN" fixed="defer"/>
      <xs:attribute name="implements" type="xh11d:URIorSafeCURIEs"/>
      <xs:attribute name="src" type="xs:anyURI"/>
      <xs:attribute name="type" type="xh11d:ContentTypes"/>
      <xs:anyAttribute />
    </xs:attributeGroup>
    <xs:group name="xmlScripting.script.content">
        <xs:sequence/>
    </xs:group>
    <xs:complexType name="xmlScripting.script.type" mixed="true">
        <xs:group ref="xmlScripting.script.content"/>
        <xs:attributeGroup ref="xmlScripting.script.attlist"/>
    </xs:complexType>
  </xs:schema>
```

F. XHTML 2.0 Document Type Definition

This appendix is *normative*.

This appendix will contain the implementation of the XHTML 2.0 DTD driver file and content model file.

G. XHTML DTD Module Implementations

This appendix is *normative*.

This appendix will contain implementations of the modules defined in this specification. These module implementations can be used in other XHTML Family Document Types.

G.1. XHTML Modular Framework

In order to take advantage of the XHTML DTD Modules, DTD authors need to define the content model for their DTD. XHTML provides a variety of tools to ease this effort. They are defined in a set of support modules, instantiated by a main Framework module:

Note that the module above references a content model module. This module is defined on a per-document type basis in addition to the document type driver file.

G.2. XHTML Module Implementations

This section will contain the formal definition of each of the XHTML Abstract Modules as a DTD module.

G.3. XHTML DTD Support Modules

The modules in this section are elements of the XHTML DTD implementation that, while hidden from casual users, are important to understand when creating derivative markup languages using the Modularization architecture.

H. Style sheet for XHTML 2

This appendix is *normative*.

This Appendix defines a normative [CSS2 [p.309]] style sheet for XHTML 2. While visual user agents implementing XHTML 2 are not required to support CSS2, their default behavior should be as if the following CSS2 styles are in effect. Note that this default behavior can be overridden by any number of things - including, for example, user-defined preferences.

```
@namespace url("http://www.w3.org/2002/06/xhtml2/");
/*  A sample style sheet for XHTML 2.0
    This style sheet describes a very incomplete, sample rendering of
    XHTML 2.0 elements.
    Editor: Masayasu Ishikawa <mimasa@w3.org>
    Revision: $Id: xhtml2.ps,v 1.3 2010/12/16 20:46:31 plehegar Exp $
*/
/* new elements */
section, h, nl, label, l, blockcode, separator, di
                { display: block; }
section, h, nl, label, l, blockcode, di
                { unicode-bidi: embed }
nl              { margin: 1.33em 0 }
summary, standby, handler
                { display: none }
blockcode       { font-family: monospace; white-space: pre }
separator       { border-bottom: thin black solid; border: 1px;
                  inset; width 100%}
h               { display: block; font-weight: bolder; font-family: sans-serif }
h1, h2, h3, h4, h5, h6
                { font-family: sans-serif; font-weight: bolder }
body h, h1 {
  font-size: 2em;
  margin: .67em 0;
}
section h, h2 {
  font-size: 1.5em;
  margin: .83em 0;
}
section section h, h3 {
  font-size: 1.17em;
  margin: 1em 0;
}
section section section h, h4, p, blockquote, ul, ol, dl
                { margin: 1.33em 0; }
section section section section h, h5 {
  font-size: .83em;
  line-height: 1.17em;
  margin: 1.67em 0;
}
section section section section section h, h6 {
  font-size: .67em;
  margin: 2.33em 0;
}
*[edit="deleted"]  { display: none }
/* no special presentation by default
```

```
*[edit="inserted"] {  }
*[edit="changed"]  {  }
*[edit="moved"]    {  }
*/
/* experimental navigation list style */
nl {
  height: 1.5em;
  overflow: hidden;
  margin: 0;
  line-height: normal !important;
  white-space: nowrap;
  text-align: start;
  cursor: default;
  border-width: 2px !important;
  border-style: inset !important;
  vertical-align: baseline;
  padding: 0;
}
nl:hover { height: auto; overflow: visible; }
nl > li, nl > label {
  display: block;
  min-height: 1em;
  line-height: normal !important;
}
nl > li, nl > label {
  padding: 0 5px 0 3px;
}
nl > li {
  margin-left: 1em;
}
nl > label {
  font-weight: bold;
}
nl > nl > label {
  display: block;
  line-height: normal !important;
  font-style: italic;
  font-weight: bold;
}
nl > nl > li {
  padding-left: 2em;
  font-style: normal;
  font-weight: normal;
}
/* inherited elements */
html, body, div, p, h1, h2, h3, h4, h5, h6,
address, blockquote, pre, ol, ul, dl, dt, dd
                { display: block }
li              { display: list-item }
head, style, link, meta
                { display: none }
table           { display: table;
                  border-spacing: 0;
                  border-top: thin black solid;
                  border-left: thin black solid }
tr              { display: table-row }
thead           { display: table-header-group }
```

```
tbody           { display: table-row-group }
tfoot           { display: table-footer-group }
col             { display: table-column }
colgroup        { display: table-column-group }
td, th          { display: table-cell;
                  border-right: thin black solid;
                  border-bottom: thin black solid;
                  padding 2px }
caption         { display: table-caption }
table:hover summary { display: block }
th              { font-weight: bolder; text-align: center }
caption         { text-align: center }
body            { padding: 8px; line-height: 1.2 }
strong          { font-weight: bolder }
blockquote      { margin-left: 4em; margin-right: 4em }
cite, em, q, var, address
                { font-style: italic }
pre code, kbd, samp
                { font-family: monospace }
pre             { white-space: pre }
sub, sup        { font-size: smaller }
sub             { vertical-align: sub }
sup             { vertical-align: super }
ol, ul, dd      { margin-left: 4em }
ol              { list-style-type: decimal }
ol ul, ul ol, ul ul, ol ol
                { margin-top: 0; margin-bottom: 0 }
abbr[title]     { border-bottom: dotted 1px }
:link           { text-decoration: underline; color: blue; }
:focus          { outline: thin dotted invert }
/* Hover effects should be default */
:link:hover,:link:visited   { color: #b7f }
/* begin bidirectionality settings (do not change) */
*[dir="ltr"]    { direction: ltr; unicode-bidi: embed }
*[dir="rtl"]    { direction: rtl; unicode-bidi: embed }
*[dir="lro"]    { direction: ltr; unicode-bidi: bidi-override }
*[dir="rlo"]    { direction: rtl; unicode-bidi: bidi-override }
/* block-level elements */
body, div, p, hr, h1, h2, h3, h4, h5, h6,
address, blockquote, pre, ol, ul, li, di, dt, dd,
table, thead, tbody, tfoot, tr, td, th,
col, colgroup, caption, object, summary, standby, blockcode
                { unicode-bidi: embed }
/* end bidi settings */
/* end xhtml2.css */
```

I. List of Elements

This appendix is *informative*.

This appendix will contain a list of elements defined in this specification, sorted in alphabetical order, with some other relevant information and links to the element definitions.

Element Name	Module	Description
a [p.55]	XHTML Hypertext Module [p.55]	Define an anchor or a link
abbr [p.48]	XHTML Text Module [p.47]	An abbreviation
access [p.73]	XHTML Access Module [p.73]	Define an accessibility mapping to an element
action [p.149]	XML Handlers Module [p.149]	No title
action [p.143]	XForms Module [p.143]	XForms Action
addEventListener [p.149]	XML Handlers Module [p.149]	No title
address [p.40]	XHTML Structural Module [p.39]	Define an address
blockcode [p.40]	XHTML Structural Module [p.39]	Define a block of computer code
blockquote [p.41]	XHTML Structural Module [p.39]	Define a large quotation
body [p.37]	XHTML Document Module [p.35]	Content of the document
br [p.48]	XHTML Text Module [p.47]	A line break
caption [p.81]	XHTML Caption Module [p.81]	Set the caption for a table
cite [p.49]	XHTML Text Module [p.47]	A citation
code [p.49]	XHTML Text Module [p.47]	A code fragment
col [p.122]	XHTML Tables Module [p.121]	Define attributes for a column
colgroup [p.122]	XHTML Tables Module [p.121]	Define attributes for a group of columns
dd [p.58]	XHTML List Module [p.57]	Definition Data
del [p.153]	XHTML Legacy Edit Module [p.153]	Indicate something was deleted
delete [p.143]	XForms Module [p.143]	XForms Delete
dfn [p.50]	XHTML Text Module [p.47]	A definition
di [p.58]	XHTML List Module [p.57]	Definition Item
dispatchEvent [p.149]	XML Handlers Module [p.149]	No title
dispatch [p.143]	XForms Module [p.143]	XForms Dispatch
div [p.41]	XHTML Structural Module [p.39]	Define the characteristics of a block

dl [p.58]	XHTML List Module [p.57]	Definition List
dt [p.58]	XHTML List Module [p.57]	Definition Term
em [p.50]	XHTML Text Module [p.47]	Emphasis
group [p.143]	XForms Module [p.143]	XForms element group
head [p.36]	XHTML Document Module [p.35]	Document metadata
heading [p.42]	XHTML Structural Module [p.39]	Set a heading
html [p.36]	XHTML Document Module [p.35]	Document root
img [p.87]	XHTML Image Module [p.87]	Incorporate an image
input [p.143]	XForms Module [p.143]	XForms Input
ins [p.154]	XHTML Legacy Edit Module [p.153]	Indicate something was inserted
insert [p.143]	XForms Module [p.143]	XForms Insert
kbd [p.50]	XHTML Text Module [p.47]	User input
l [p.51]	XHTML Text Module [p.47]	A line of text
legacyheadings [p.155]	XHTML Legacy Headings Module [p.155]	Set a heading
li [p.60]	XHTML List Module [p.57]	List item
link [p.95]	XHTML Metainformation Module [p.95]	A link to another resource
listener [p.147]	XML Events Module [p.147]	Event listener
load [p.143]	XForms Module [p.143]	XForms Load
message [p.143]	XForms Module [p.143]	XForms Message
meta [p.97]	XHTML Metainformation Module [p.95]	Meta information
model [p.143]	XForms Module [p.143]	XForms Model
object [p.101]	XHTML Object Module [p.101]	External object
ol [p.59]	XHTML List Module [p.57]	Ordered list
output [p.143]	XForms Module [p.143]	XForms Output
p [p.43]	XHTML Structural Module [p.39]	Define a paragraph
param [p.106]	XHTML Object Module [p.101]	Parameter for external object
pre [p.43]	XHTML Structural Module [p.39]	Define a preformatted block
preventDefault [p.149]	XML Handlers Module [p.149]	No title
q [p.51]	XHTML Text Module [p.47]	A quotation
range [p.143]	XForms Module [p.143]	XForms range definition

rebuild [p.143]	XForms Module [p.143]	XForms Rebuild
recalculate [p.143]	XForms Module [p.143]	XForms Recalculate
refresh [p.143]	XForms Module [p.143]	XForms Refresh
removeEventListener [p.149]	XML Handlers Module [p.149]	No title
repeat [p.143]	XForms Module [p.143]	XForms repeating group
reset [p.143]	XForms Module [p.143]	XForms Reset
revalidate [p.143]	XForms Module [p.143]	XForms Revalidate
ruby [p.113]	Ruby Module [p.113]	Ruby Annotation
samp [p.52]	XHTML Text Module [p.47]	Sample output
script [p.151]	XML Scripting Module [p.151]	No title
secret [p.143]	XForms Module [p.143]	XForms Secret Input
section [p.44]	XHTML Structural Module [p.39]	Define a section of a document
select1 [p.143]	XForms Module [p.143]	XForms single select
select [p.143]	XForms Module [p.143]	XForms multiple select
send [p.143]	XForms Module [p.143]	XForms Send
separator [p.45]	XHTML Structural Module [p.39]	Insert a break in a document
setfocus [p.143]	XForms Module [p.143]	XForms Setfocus
setindex [p.143]	XForms Module [p.143]	XForms Index
setvalue [p.143]	XForms Module [p.143]	XForms Setvalue
span [p.52]	XHTML Text Module [p.47]	Define characteristics of text
standby [p.110]	XHTML Object Module [p.101]	Message to render while loading object
stopPropagation [p.149]	XML Handlers Module [p.149]	No title
strong [p.53]	XHTML Text Module [p.47]	Strong emphasis
style [p.115]	XHTML Style Sheet Module [p.115]	Definition of style rules
sub [p.53]	XHTML Text Module [p.47]	A subscript
submit [p.143]	XForms Module [p.143]	XForms submit
summary [p.124]	XHTML Tables Module [p.121]	Set the summary for a table
sup [p.53]	XHTML Text Module [p.47]	A superscript
switch [p.143]	XForms Module [p.143]	XForms selection
table [p.125]	XHTML Tables Module [p.121]	Define a table
tbody [p.135]	XHTML Tables Module [p.121]	Define the body for a table
td [p.135]	XHTML Tables Module [p.121]	Define a table cell

textarea [p.143]	XForms Module [p.143]	XForms Textarea
tfoot [p.140]	XHTML Tables Module [p.121]	Define the footer for a table
th [p.135]	XHTML Tables Module [p.121]	Define a table header
thead [p.140]	XHTML Tables Module [p.121]	Define the heading for a table
title [p.37]	XHTML Document Module [p.35]	Title for document
tr [p.141]	XHTML Tables Module [p.121]	Define a table row
trigger [p.143]	XForms Module [p.143]	XForms trigger
ul [p.59]	XHTML List Module [p.57]	Unordered list
upload [p.143]	XForms Module [p.143]	XForms file upload
var [p.54]	XHTML Text Module [p.47]	A variable

J. List of Attributes

This appendix is *informative*.

Attribute Name	Module	Description
abbr [p.135]	XHTML Tables Module [p.121]	Abbreviated form
about [p.99]	XHTML Metainformation Attributes Module [p.99]	Define which resource that has a specified property
activate [p.74]	XHTML Access Module [p.73]	Define action when access target obtains focus
archive [p.101]	XHTML Object Module [p.101]	Define list of object archives
axis [p.135]	XHTML Tables Module [p.121]	Define a category for a cell
charset [p.151]	XML Scripting Module [p.151]	Indicate the expected encoding of the remote resource
cite [p.65]	XHTML Hypertext Attributes Module [p.65]	Set the URI for a citation
class [p.61]	XHTML Core Attributes Module [p.61]	Set the class of an element
colspan [p.135]	XHTML Tables Module [p.121]	Set the number of columns a table cell should span
content-length [p.101]	XHTML Object Module [p.101]	Indicate the size of a link's target
content [p.99]	XHTML Metainformation Attributes Module [p.99]	Set the content for metadata
coords [p.90]	XHTML Image Map Attributes Module [p.89]	Define coordinates for am element
datatype [p.99]	XHTML Metainformation Attributes Module [p.99]	Indicate the datatype of external metadata
datetime [p.83]	XHTML Edit Attributes Module [p.83]	Define date and time
declare [p.101]	XHTML Object Module [p.101]	Indicate that an object should only be loaded
defaultAction [p.147]	XML Events Module [p.147]	Indicate the default action for the event
defer [p.151]	XML Scripting Module [p.151]	Indicate processing of contents should be defered
dir [p.79]	XHTML Bi-directional Text Attribute Module [p.79]	Define the direction of text
disabled [p.115]	XHTML Style Sheet Module [p.115]	Indicate that an item is disabled

edit [p.83]	XHTML Edit Attributes Module [p.83]	Indicate how an item was edited
encoding [p.85]	XHTML Embedding Attributes Module [p.85]	Define the encoding of an external source
eventTarget [p.147]	XML Events Module [p.147]	Set the id on which the event is monitored
event [p.147]	XML Events Module [p.147]	Set the event type
full [p.48]	XHTML Text Module [p.47]	Refer to the full version of an abbreviation
handler [p.147]	XML Events Module [p.147]	Specify the handler method for an event
headers [p.136]	XHTML Tables Module [p.121]	Define the header cells that relate to this cell
href [p.65]	XHTML Hypertext Attributes Module [p.65]	Define a URI target when the element is activated
hreflang [p.65]	XHTML Hypertext Attributes Module [p.65]	Indicate the base language of the target
hrefmedia [p.66]	XHTML Hypertext Attributes Module [p.65]	Indicate the target media of an href attribute
hreftype [p.66]	XHTML Hypertext Attributes Module [p.65]	Indicate the content type of an href attribute
id [p.62]	XHTML Core Attributes Module [p.61]	Define the ID for the element
implements [p.151]	XML Scripting Module [p.151]	Indication of what a script implements
ismap [p.90]	XHTML Image Map Attributes Module [p.89]	Indicate whether this is an imagemap
its:translate [p.71]	XHTML I18N Attribute Module [p.71]	Should an element be translated
key [p.74]	XHTML Access Module [p.73]	Define a shortcut key
layout [p.62]	XHTML Core Attributes Module [p.61]	Indicate whether whitespace is relevant
media [p.75]	XHTML Access Module [p.73]	Define the applicable media
media [p.93]	XHTML Media Attribute Module [p.93]	Define the applicable media
media [p.115]	XHTML Style Sheet Module [p.115]	Define the applicable media
name [p.106]	XHTML Object Module [p.101]	Set the name of an applet parameter

nextfocus [p.66]	XHTML Hypertext Attributes Module [p.65]	Define the order in which this control is accessed
observer [p.147]	XML Events Module [p.147]	Set the observer for an event
order [p.75]	XHTML Access Module [p.73]	Define navigation order method
phase [p.147]	XML Events Module [p.147]	Specify the phase in which the event is tracked
prevfocus [p.68]	XHTML Hypertext Attributes Module [p.65]	Define the order in which this control is accessed
profile [p.36]	XHTML Document Module [p.35]	Define the location of the metadata profiles
propagate [p.147]	XML Events Module [p.147]	Indicate whether the event should continue to propagate
property [p.99]	XHTML Metainformation Attributes Module [p.99]	Indicate the property being defined
rel [p.99]	XHTML Metainformation Attributes Module [p.99]	Define the relationship to the target
resource [p.99]	XHTML Metainformation Attributes Module [p.99]	Define the resource a property is related to
rev [p.99]	XHTML Metainformation Attributes Module [p.99]	Define the relationship from the target to subject
role [p.111]	XHTML Role Attribute Module [p.111]	Define the role of this element
rowspan [p.136]	XHTML Tables Module [p.121]	Define the number of rows a cell spans
scope [p.136]	XHTML Tables Module [p.121]	Define the scope of a header
shape [p.90]	XHTML Image Map Attributes Module [p.89]	Define the shape of a map
span [p.122]	XHTML Tables Module [p.121]	Define the number of columns a colgroup spans
src [p.86]	XHTML Embedding Attributes Module [p.85]	Define the URI for an external source for the element
srctype [p.86]	XHTML Embedding Attributes Module [p.85]	Indicate the content type of a src attribute
style [p.119]	XHTML Style Attribute Module [p.119]	Set the style to use on the element
target [p.68]	XHTML Hypertext Attributes Module [p.65]	Set the target window for a link
targetid [p.76]	XHTML Access Module [p.73]	Define the target element for an event

targetrole [p.76]	XHTML Access Module [p.73]	Define the target role(s) for an access key
title [p.62]	XHTML Core Attributes Module [p.61]	Set the title for an element
type [p.106]	XHTML Object Module [p.101]	Define the type of a referenced value element
typeof [p.99]	XHTML Metainformation Attributes Module [p.99]	Define the RDF type of metadata
usemap [p.89]	XHTML Image Map Attributes Module [p.89]	Set the name of an image map to use
value [p.60]	XHTML List Module [p.57]	Set the value for a list item
value [p.106]	XHTML Object Module [p.101]	Set the value for an applet parameter
valuetype [p.106]	XHTML Object Module [p.101]	Set the type of an applet value
version [p.36]	XHTML Document Module [p.35]	Set the XHTML document version
xml:base [p.68]	XHTML Hypertext Attributes Module [p.65]	Set the base URI for the element
xml:id [p.62]	XHTML Core Attributes Module [p.61]	Define the ID for the element
xml:lang [p.71]	XHTML I18N Attribute Module [p.71]	Set the language of the element
xsi:schemaLocation [p.36]	XHTML Document Module [p.35]	Set the location for an XML Schema

K. Cross-reference Index

This appendix is *informative*.

This appendix will contain a detailed index of this document, with links to the indexed terms.

L. References

This appendix is *normative*.

L.1. Normative References

[CSS2]
"*Cascading Style Sheets, level 2 (CSS2) Specification*", W3C Recommendation, B. Bos *et al.*, *eds.*, 12 May 1998.
Available at: http://www.w3.org/TR/1998/REC-CSS2-19980512

[CSS3-TEXT]
"*CSS3 Text Effects Module*", W3C Working Draft, E. J. Etemad, *ed.*, 27 June 2005, *work in progress*.
Available at: http://www.w3.org/TR/2005/WD-css3-text-20050627/
The latest version is available at: http://www.w3.org/TR/css3-text/

[CURIE]
"*CURIE Syntax 1.0*", W3C Candidate Recommendation, M. Birbeck *et al.*, 16 January 2009.
Available at: http://www.w3.org/TR/2009/CR-curie-20090116

[DCORE]
"*DCMI Metadata Terms*", 13 June 2005.
Available at: http://dublincore.org/documents/dcmi-terms/

[DOM]
"*Document Object Model (DOM) Level 2 Core Specification*", W3C Recommendation, A. Le Hors *et al.*, *eds.*, 13 November 2000.
Available at: http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113
A list of DOM Level 2 specifications can be found at: http://www.w3.org/DOM/DOMTR#dom2

[DOMEVENTS]
"*Document Object Model (DOM) Level 2 Events Specification*", W3C Recommendation, T. Pixley, ed., 13 November 2000.
Available at: http://www.w3.org/TR/2000/REC-DOM-Level-2-Events-20001113
A list of DOM Level 2 specifications can be found at: http://www.w3.org/DOM/DOMTR#dom2

[IRI]
"*Internationalized Resource Identifiers (IRIs)*", RFC 3987, M. Dürst and M. Suignard, January 2005.
Available at: http://www.rfc-editor.org/rfc/rfc3987.txt

[ITS]
"*The Internationalization Tag Set (ITS) Version 1.0*", W3C Recommendation, C. Lieske, F. Sasaki, 3 April 2007.
Available at: http://www.w3.org/TR/2007/REC-its-20070403/

[MIMETYPES]
List of registered content types (MIME media types). Download a list of registered content types from http://www.iana.org/assignments/media-types/.

[P3P]
"*The Platform for Privacy Preferences 1.0 (P3P1.0) Specification*", W3C Recommendation, L. Cranor *et al.*, 16 April 2002.
Available at: http://www.w3.org/TR/2002/REC-P3P-20020416/
[RDFASYNTAX]
"*RDFa in XHTML: Syntax and Processing*", W3C Recommendation, J. Miller *et al.*, 14 October 2008.
Available at: http://www.w3.org/TR/2008/REC-rdfa-syntax-20081014
[RELAXNG]
"*RELAX NG Specification*", OASIS Committee Specification, J. Clark, Murata M., *eds.*, 3 December 2001.
Available at: http://relaxng.org/spec-20011203.html
RELAX NG has been standardized as part of ISO/IEC 19757 - Document Schema Definition Languages (DSDL), as ISO/IEC 19757-2:2003 "Information technology -- Document Schema Definition Language (DSDL) -- Part 2: Regular-grammar-based validation -- RELAX NG". See home page for Document Schema Definition Languages at http://dsdl.org/ for details.
[RFC2045]
"*Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*", RFC 2045, N. Freed and N. Borenstein, November 1996.
Available at: http://www.rfc-editor.org/rfc/rfc2045.txt
[RFC2119]
"*Key words for use in RFCs to indicate requirement levels*", RFC 2119, S. Bradner, March 1997.
Available at: http://www.rfc-editor.org/rfc/rfc2119.txt
[RFC2616]
"*Hypertext Transfer Protocol -- HTTP/1.1*", RFC 2616, R. Fielding *et al.*, June 1999.
Available at: http://www.rfc-editor.org/rfc/rfc2616.txt
[RFC3066]
"*Tags for the Identification of Languages*", RFC 3066, H. Alvestrand, January 2001.
Available at: http://www.rfc-editor.org/rfc/rfc3066.txt
[RUBY]
"*Ruby Annotation*", W3C Recommendation, M. Sawicki *et al.*, *eds.*, 31 May 2001.
Available at: http://www.w3.org/TR/2001/REC-ruby-20010531
[SGML]
"*Information Processing -- Text and Office Systems -- Standard Generalized Markup Language (SGML)*", ISO 8879:1986.
Please consult the ISO Web site at http://www.iso.org/ for information about the standard, or http://www.oasis-open.org/cover/general.html#overview about SGML.
[UAAG1]
"*User Agent Accessibility Guidelines 1.0*". Ian Jacobs *et al.*, 17 December 2002.
Available at: http://www.w3.org/TR/2002/REC-UAAG10-20021217
The latest version is available at: http://www.w3.org/TR/UAAG10
[UAX9]
"*Unicode Standard Annex #9: The Bidirectional Algorithm*", M. Davis, 25 March 2005.
Available at: http://www.unicode.org/reports/tr9/tr9-15.html

The latest version of UAX #9 is available at: http://www.unicode.org/reports/tr9/

[URI]
"*Uniform Resource Identifiers (URI): Generic Syntax*", RFC 3986, T. Berners-Lee *et al.*, January 2005.
Available at: http://www.rfc-editor.org/rfc/rfc3986.txt.

[XFORMS]
"*XForms 1.1*", W3C Candidate Recommendation, John Boyer, *eds.*, 29 November 2007.
Available at: http://www.w3.org/TR/2007/CR-xforms11-20071129/

[XHTMLACCESS]
"*XHTML Access Module*", W3C Working Draft, M. Birbeck *et al.*, 26 May 2008.
Available at: http://www.w3.org/TR/2008/WD-xhtml-access-20080526

[XHTMLMOD]
"*Modularization of XHTML*", W3C Recommendation, M. Altheim *et al.*, *eds.*, 10 April 2001
Available at: http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410

[XHTMLROLE]
"*XHTML Role Attribute Module*", W3C Working Draft , M. Birbeck *et al.*, 7 April 2008.
Available at: http://www.w3.org/TR/2008/WD-xhtml-role-20080407

[XHTMLVOCAB]
"*XHTML Vocabulary*", Steven Pemberton, 21 October 2008.
Available at: http://www.w3.org/1999/xhtml/vocab/

[XML]
"*Extensible Markup Language (XML) 1.0 (Third Edition)*", W3C Recommendation, T. Bray *et al.*, *eds.*, 4 February 2004.
Available at: http://www.w3.org/TR/2004/REC-xml-20040204

[XMLBASE]
"*XML Base*", W3C Recommendation, J. Marsh, *ed.*, 27 June 2001.
Available at: http://www.w3.org/TR/2001/REC-xmlbase-20010627/

[XMLEVENTS]
"*XML Events 2*", W3C Working Draft, S. McCarron *et al.*, *eds.*, 16 February 2007.
Available at: http://www.w3.org/TR/2007/WD-xml-events-20070216

[XMLID]
"*xml:id Version 1.0*", W3C Recommendation, J. Marsh, D. Veillard, N. Walsh, *eds.*, 9 September 2005.
Available at: http://www.w3.org/TR/2005/REC-xml-id-20050909/

[XMLNS]
"*Namespaces in XML*", W3C Recommendation, T. Bray *et al.*, *eds.*, 14 January 1999.
Available at: http://www.w3.org/TR/1999/REC-xml-names-19990114

[XMLSCHEMA]
"*XML Schema Part 1: Structures Second Edition*", W3C Recommendation, H. S. Thompson *et al.*, *eds.*, 28 October 2004.
Available at: http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/
"*XML Schema Part 2: Datatypes Second Edition*", W3C Recommendation, P. V. Biron, A. Malhotra, *eds.*, 28 October 2004.
Available at: http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/

L.2. Informative References

[HTML4]
"*HTML 4.01 Specification*", W3C Recommendation, D. Raggett *et al.*, *eds.*, 24 December 1999.
Available at: http://www.w3.org/TR/1999/REC-html401-19991224

[RDF]
"*Resource Description Framework (RDF): Concepts and Abstract Syntax* ", W3C Recommendation, G. Klyne, J. Carrol, *ed.*, 10 February 2004.
Available at: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

[SVG]
"*Scalable Vector Graphics (SVG) 1.1 Specification*", Jon Ferraiolo *et al.*
Available at: http://www.w3.org/TR/SVG/

[XFRAMES]
"*XFrames*", W3C Working Draft, S. Pemberton, *ed.*, 6 August 2002, *work in progress*.
Available at: http://www.w3.org/TR/2002/WD-xframes-20020806

[XLINK]
"*XML Linking Language (XLink) Version 1.0*", W3C Recommendation, S. DeRose *et al.*, *eds.*, 27 June 2001.
Available at: http://www.w3.org/TR/2001/REC-xlink-20010627/

[XMLSTYLE]
"*Associating Style Sheets with XML documents Version 1.0*", W3C Recommendation, J. Clark, *ed.*, 29 June 1999.
Available at: http://www.w3.org/1999/06/REC-xml-stylesheet-19990629

[XPATH]
"*XML Path Language (XPath) Version 1.0*", W3C Recommendation, J. Clark *et al.*, *eds.*, 16 November 1999.
Available at: http://www.w3.org/TR/1999/REC-xpath-19991116

M. Acknowledgements

This appendix is *informative*.

This document was prepared by the W3C XHTML2 Working Group. The members at the time of publication of the Note were:

- Markus Gylling, DAISY Consortium (XHTML 2 Working Group Co-Chair)
- Steven Pemberton, CWI (XHTML 2 Working Group Co-Chair)
- Mark Birbeck, Sidewinder Labs (Invited Expert)
- Susan Borgrink, Progeny Systems
- Christina Bottomley, Society for Technical Communication (STC)
- Alessio Cartocci, International Webmasters Association / HTML Writers Guild (IWA-HWG)
- Alexander Graf, University of Innsbruck
- Tina Holmboe, Greytower Technologies (Invited Expert)
- John Kugelman, Progeny Systems
- Luca Mascaro, International Webmasters Association / HTML Writers Guild (IWA-HWG)
- Shane McCarron, Applied Testing and Technology, Inc. (Invited Expert)
- Michael Rawling, IVIS Group Limited
- Gregory Rosmaita, Invited Expert
- Sebastian Schnitzenbaumer, Dreamlab Technologies AG
- Richard Schwerdtfeger, IBM
- Elias Torres, IBM
- Masataka Yakura, Mitsue-Links Co., Ltd.
- Toshihiko Yamakami, ACCESS Co., Ltd.

The HTML Working Group would like to acknowledge the great many people outside of the HTML Working Group who help with the process of developing the XHTML 2.0 specification. These people are too numerous to list individually. They include but are not limited to people who have contributed on the www-html@w3.org mailing list, other Working Groups at the W3C, and the W3C Team. XHTML 2.0 is truly a cooperative effort between the HTML Working Group, the rest of the W3C, and the public.