Modularization of XHTML ™

W3C Working Draft4 October 2000

This version:
http://www.w3.org/TR/2000/WD-xhtml-modularization-20001004
(Single HTML file [p.1] , Postscript version, PDF version, ZIP archive, or Gzip'd TAR archive)
Latest version:
http://www.w3.org/TR/xhtml-modularization
Previous version:
http://www.w3.org/TR/2000/WD-xhtml-modularization-20000105/
Diff-marked version:
xhtml-modularization-diff-20001004.html

Editors:
Robert Adams, Intel Corporation
Murray Altheim, Sun Microsystems
Frank Boumphrey, HTML Writers Guild
Sam Dooley, IBM
Shane McCarron, Applied Testing and Technology
Sebastian Schnitzenbaumer, Mozquito Technologies
Ted Wugofski, Phone.com (formerly Gateway)

Abstract

This Working Draft specifies an abstract modularization of XHTML and an implementation of the abstraction using XML Document Type Definitions (DTDs). This modularization provide a means for subsetting and extending XHTML, a feature needed for extending XHTML's reach onto emerging platforms.

Status of this document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest status of this document series is maintained at the W3C.

This is the "Working Draft" of "Modularization of XHTML". It is a version that incorporates some comments from the Last Call Working Draft review period. A diff-marked version from the Last Call draft is available for comparison purposes.

Major changes in this version include:

- Re-integration of the Building document into this document
- Incorporation of the Henry Thompson/Dan Connolly XML Namespace handling process with substantial additions by the Math and HTML working groups
- Complete worked examples including modules and miniature DTDs.
- Minor restructuring of abstract module definitions, including the creation of a "style attribute module", a "name identification module" and addition of a "target" module.
- Tweaking of some of the module contents based on review comments

Please send review comments to www-html-editor@w3.org.

The Working Group anticipates asking the W3C Director to advance this document to Candidate Recommendation after the Working Group processes Last Call review comments and incorporates resolutions into the Guidelines.

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

This is a W3C Working Draft for review by W3C Members and other interested parties. It is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to use W3C Working Drafts as reference material or to cite them as other than "work in progress". This is work in progress and does not imply endorsement by, or the consensus of, either W3C or participants of the HTML WG Group.

A list of current W3C Recommendations and other technical documents can be found at http://www.w3.org/TR.

Public discussion on HTML features takes place on the mailing list www-html@w3.org (archive). The W3C staff contact for work on HTML is Dave Raggett.

Quick Table of Contents

Full Table of Contents

1. Introduction

This section is *informative*.

1.1. What is XHTML?

XHTML is the reformulation of HTML 4 as an application of XML. XHTML 1.0 [XHTML1] [p.168] specifies three XML document types that correspond to the three HTML 4 DTDs: Strict, Transitional, and Frameset. XHTML 1.0 is the basis for a family of document types that subset and extend HTML.

1.2. What is XHTML Modularization?

XHTML Modularization is a decomposition of XHTML 1.0, and by reference HTML 4, into a collection of abstract modules that provide specific types of functionality. These abstract modules are implemented in this specification using the XML Document Type Definition language, but an implementation using XML Schemas is expected. The rules for defining the abstract modules, and for implementing them using XML DTDs, are also defined in this document.

These modules may be combined with each other and with other modules to create XHTML subset and extension document types that qualify as members of the XHTML family of document types.

1.3. Why Modularize XHTML?

The modularization of XHTML refers to the task of specifying well-defined sets of XHTML elements that can be combined and extended by document authors, document type architects, other XML standards specifications, and application and product designers to make it economically feasible for content developers to deliver content on a greater number and diversity of platforms.

Over the last couple of years, many specialized markets have begun looking to HTML as a content language. There is a great movement toward using HTML across increasingly diverse computing platforms. Currently there is activity to move HTML onto mobile devices (hand held computers, portable phones, etc.), television devices (digital televisions, TV-based web browsers, etc.), and appliances (fixed function devices). Each of these devices has different requirements and constraints.

Modularizing XHTML provides a means for product designers to specify which elements are supported by a device using standard building blocks and standard methods for specifying which building blocks are used. These modules serve as "points of conformance" for the content community. The content community can now target the installed base that supports a certain collection of modules, rather than worry about the installed base that supports this permutation of XHTML elements or that permutation of XHTML elements. The use of standards is critical for modularized XHTML to be successful on a large scale. It is not economically feasible for content developers to tailor content to each and every permutation of XHTML elements. By specifying a standard, either software processes can autonomously tailor content to a device, or the device can automatically load the software required to process a module.

Modularization also allows for the extension of XHTML’s layout and presentation capabilities, using the extensibility of XML, without breaking the XHTML standard. This development path provides a stable, useful, and implementable framework for content developers and publishers to manage the rapid pace of technological change on the Web.

1.3.1. Abstract modules

An XHTML document type is defined as a set of abstract modules. A abstract module defines one kind of data that is semantically different from all others. Abstract modules can be combined into document types without a deep understanding of the underlying schema that define the modules.

1.3.2. Module implementations

A module implementation consists of a set of element types, a set of attribute list declarations, and a set of content model declarations, where any of these three sets may be empty. An attribute list declaration in a module may modify an element type outside the element types defined in the module, and a content model declaration may modify an element type outside the element type set of the module.

One implementation mechanism is XML DTDs. An XML DTD is a means of describing the structure of a class of XML documents, collectively known as an XML document type. XML DTDs are described in the XML 1.0 Recommendation [XML] [p.168] . Another implementation mechanism is XML Schema [XMLSCHEMA] [p.168] .

1.3.3. Hybrid document types

A hybrid document type is an document type composed from a collection of XML DTDs or DTD Modules. The primary purpose of the modularization framework described in this document is to allow a DTD author to combine elements from multiple abstract modules into a hybrid document type, develop documents against that hybrid document type, and to validate that document against the associated hybrid document type definition.

One of the most valuable benefits of XML over SGML is that XML reduces the barrier to entry for standardization of element sets that allow communities to exchange data in an interoperable format. However, the relatively static nature of HTML as the content language for the Web has meant that any one of these communities have previously held out little hope that their XML document types would be able to see widespread adoption as part of Web standards. The modularization framework allows for the dynamic incorporation of these diverse document types within the XHTML family of document types, further reducing the barriers to the incorporation of these domain-specific vocabularies in XHTML documents.

1.3.4. Validation

The use of well-formed, but not valid, documents is an important benefit of XML. In the process of developing a document type, however, the additional leverage provided by a validating parser for error checking is important. The same statement applies to XHTML document types with elements from multiple abstract modules.

A document is an instance of one particular document type; defined by the DTD identified in the document’s prologue. Validating the document is the process of checking that the document complies with the rules in the document type definition.

One document can consist of multiple document fragments. Validating only fragments of a document, where each fragment is of a different document type than the other fragments in the document, is beyond the scope of this framework - since it would require technology that is not yet defined.

However, the modularization framework allows multiple document type definitions to be integrated and form a new document type (e.g. SVG integrated into XHTML). The new document type definition can be used for normal XML 1.0 validation.

1.3.5. Formatting Model

Earlier versions of HTML attempted to define parts of the model that user agents are required to use when formatting a document. With the advent of HTML 4, the W3C started the processing of divorcing presentation from structure. XHTML 1.0 maintained this separation, and this document continues moving HTML and its descendants down this path. Consequently, this document makes no requirements on the formatting model associated with the presentation of documents marked up with XHTML Family document types.

Instead, this document recommends that content authors rely upon style mechanisms such as CSS to define the formatting model for their content. When user agents support the style mechanisms, documents will format as expected. When user agents do not support the style mechanisms, documents will format as appropriate for that user agent. This permits XHTML Family user agents to support rich formatting models on devices where that is appropriate, and lean formatting models on devices where *that* is appropriate.

2. Terms and Definitions

This section is *informative*.

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.168] 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).

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.167] 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 parse-able 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 or numeric string used as a reference to the content of a declared entity (eg., "&amp;" for "&", "&#60;" for "<", "&copy;" for "©".)

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.

3. Conformance Definition

This section is *normative*.

In order to ensure that XHTML-family documents are maximally portable among XHTML-family user agents, this specification rigidly defines conformance requirements for both of these and for XHTML-family document types. While the conformance definitions can be found in this section, they necessarily reference normative text within this document, within the base XHTML specification [XHTML1] [p.168] , and within other related specifications. It is only possible to fully comprehend the conformance requirements of XHTML through a complete reading of all normative references.

3.1. XHTML Host Language Document Type Conformance

It is possible to modify existing document types and define wholly new document types using both modules defined in this specification and other modules. Such a document type is "XHTML Host Language Conforming" when it meets the following criteria:

1. The document type must be defined using one of the implementation methods defined by the W3C (currently this is limited to XML DTDs, but XML Schema will be available soon).
2. The document type must have a unique identifier as defined in Naming Rules [p.18] that uses the string "XHTML" in its first token of the public text description.
3. The document type must include, at a minimum, the Structure, Hypertext, Basic Text, and List modules defined in this specification.
4. For each of the W3C-defined modules that are included, all of the elements, attributes, types of attributes (including any required enumerated value lists), and any required minimal content models must be included (and optionally extended) in the document type’s content model. When content models are extended, all of the elements and attributes (along with their types or any required enumerated value lists) required in the original content model must continue to be required.
5. The document type may define additional elements and attributes. However, these must be in their own XML Namespace [XMLNAMES] [p.168] .

3.2. XHTML Integration Set Document Type Conformance

It is also possible to define document types that are based upon XHTML, but do not adhere to its structure. Such a document type is "XHTML Integration Set Conforming" when it meets the following criteria:

1. The document type must be defined using one of the implementation methods defined by the W3C (currently this is limited to XML DTDs, but XML Schema will be available soon).
2. The document type must have a unique identifier as defined in Naming Rules [p.18] that uses the string "XHTML" NOT in its first token of the public text description.
3. The document type must include, at a minimum, the Hypertext, Basic Text, and List modules defined in this specification.

4. For each of the W3C-defined modules that are included, all of the elements, attributes, types of attributes (including any required enumerated lists), and any required minimal content models must be included (and optionally extended) in the document type’s content model. When content models are extended, all of the elements and attributes (along with their types or any required enumerated value lists) required in the original content model must continue to be required.
5. The document type may define additional elements and attributes. However, these must be in their own XML Namespace [XMLNAMES] [p.168] .

3.3. XHTML Family Module Conformance

This specification defines a method for defining XHTML-conforming modules. A module conforms to this specification when it meets all of the following criteria:

1. The module must be defined using one of the implementation methods identified in this specification (currently only XML DTDs are defined).
2. The module must have a unique identifier as defined in Naming Rules [p.18] .
3. When the module is implemented using an XML DTD, the module must insulate its parameter entity names through the use of unique prefixes or other, similar methods.
4. The module must have a prose definition that describes the syntactic and semantic requirements of the elements, attributes, and/or content models that it declares.
5. The module must not reuse any element names that are defined in other W3C-defined modules, except when the content model and semantics of those elements are either identical to the original or an extension of the original, or when the reused element names are within their own namespace (see below).
6. The module’s elements and attributes must be part of an XML Namespace [XMLNAMES] [p.168] . If the module is defined by an organization other than the W3C, this namespace must NOT be the same as the namespace in which other W3C modules are defined.

3.4. XHTML Family Document Conformance

A conforming XHTML family document is a valid instance of a conforming XHTML Host Language Conforming Document Type.

3.5. XHTML Family User Agent Conformance

A conforming user agent must meet all of the following criteria (as defined in [XHTML1] [p.168]):

1. In order to be consistent with the XML 1.0 Recommendation [XML] [p.168] , the user agent must parse and evaluate an XHTML document for well-formedness. If the user agent claims to be a validating user agent, it must also validate documents against their referenced DTDs according to [XML] [p.168] .
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. When a user agent processes an XHTML document as generic XML, it shall only recognize attributes of type `ID` (e.g., the `id` attribute on most XHTML elements) as fragment identifiers.
4. If a user agent encounters an element it does not recognize, it must continue to process the children of that element. If the content is text, the text must be presented to the user.
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. If it encounters an entity reference (other than one of the predefined entities) for which the User Agent has processed no declaration (which could happen if the declaration is in the external subset which the User Agent hasn’t read), the entity reference should be rendered as the characters (starting with the ampersand and ending with the semi-colon) that make up the entity reference.
8. 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.
9. The use agent must process whitespace characters according to the following rules. The following characters are defined in [XML] as whitespace characters:
    - Space (&#x0020;)
    - Tab (&#x0009;)
    - Carriage return (&#x000D;)
    - Line feed (&#x000A;)

    The XML processor normalizes different system’s line end codes into one single line-feed character, that is passed up to the application. The XHTML user agent in addition, must treat the following characters as whitespace:

    - Zero-width space (&#x200B;)

    Whitespace is handled according to the following rules:

    - All whitespace surrounding block elements should be removed.
    - Comments are removed entirely and do not affect whitespace handling. One whitespace character on either side of a comment is treated as two whitespace characters.
    - Leading and trailing whitespace inside a block element must be removed.
    - Line feed characters within a block element must be converted into a space (except when the ’xml:space’ attribute is set to ’preserve’).
    - A sequence of whitespace characters must be reduced to a single space character (except when the ’xml:space’ attribute is set to ’preserve’).
    - With regard to rendition, the User Agent should render the content in a manner appropriate to the language in which the content is written. In languages whose primary script is Latinate, the ASCII space character is typically used to encode both grammatical word boundaries and typographic whitespace; in languages whose script is related to Nagari (e.g., Sanskrit, Thai, etc.), grammatical boundaries may be

> encoded using the ZW ’space’ character, but will not typically be represented by typographic whitespace in rendered output; languages using Arabiform scripts may encode typographic whitespace using a space character, but may also use the ZW space character to delimit ’internal’ grammatical boundaries (what look like words in Arabic to an English eye frequently encode several words, e.g., ’kitAbuhum’ = ’kitAbu-hum’ = ’book them’ == their book); and languages in the Chinese script tradition typically neither encode such delimiters nor use typographic whitespace in this way.

Whitespace in attribute values is processed according to [XML] [p.168] .

3.6. Naming Rules

XHTML host language document types must adhere to strict naming conventions so that it is possible for software and users to readily determine the relationship of document types to XHTML. The names for document types implemented as XML Document Type Definitions are defined through Formal Public Identifiers (FPIs). Within FPIs, fields are separated by double slash character sequences (`//`). The various fields MUST be composed as follows:

1. The leading field must be "-" to indicate a privately defined resource.
2. The second field MUST contain the name of the organization responsible for maintaining the named item. There is no formal registry for these organization names. Each organization SHOULD define a name that is unique. The name used by the W3C is, for example, `W3C`.
3. The third field MUST begin with the token XHTML if the document type is Host Language conforming. It may contain the string XHTML anywhere but the first token if the document type is Integration Set conforming. No other document types may use the token XHTML in their third field. The field must also contain an organization-defined unique identifier (e.g., MyML 1.0). This identifier SHOULD be composed of a unique name and a version identifier that can be updated as the document type evolves.
4. The fourth field defines the language in which the item is developed (e.g., `EN`).

Using these rules, the name for an XHTML Host Language conforming document type might be `-//MyCompany//XHTML DTD MyML 1.0//EN`. The name for an XHTML family conforming module might be `-//MyCompany//XHTML ELEMENTS MyElements 1.0//EN`. The name for an XHTML Integration Set conforming document type might be `-//MyCompany//Special Markup with XHTML//EN`.

4. Defining Abstract Modules

This section is *normative*.

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.

4.1. 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.

`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.

4.2. Content Types

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. 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.

4.3. 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
NAME	A name with the same character constraints as ID above
NMTOKEN	A name composed of only name tokens as defined in XML 1.0 [XML] [p.168]
NMTOKENS	One or more whitespace separated NMTOKEN values
PCDATA	Processed character data

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 from [ISO10646] [p.167] .
Charset	A character encoding, as per [RFC2045] [p.167] .
Charsets	A space-separated list of character encodings, as per [RFC2045] [p.167] .
Color	The attribute value type "Color" refers to color definitions as specified in [SRGB] [p.168] . A color value may either be a hexadecimal number (prefixed by a hash mark) or one of the following sixteen color names. The color names are case-insensitive. **Color names and sRGB values** ⬛ Black = "#000000" 🟩 Green = "#008000" ⬜ Silver = "#C0C0C0" 🟩 Lime = "#00FF00" ⬛ Gray = "#808080" 🟫 Olive = "#808000" ⬜ White = "#FFFFFF" 🟨 Yellow = "#FFFF00" 🟥 Maroon = "#800000" 🟦 Navy = "#000080" 🟥 Red = "#FF0000" 🟦 Blue = "#0000FF" 🟪 Purple = "#800080" 🟦 Teal = "#008080" 🟪 Fuchsia = "#FF00FF" 🟦 Aqua = "#00FFFF" Thus, the color values "#800080" and "Purple" both refer to the color purple.
ContentType	A media type, as per [RFC2045] [p.167] .
ContentTypes	A comma-separated list of media types, as per [RFC2045] [p.167] .
Datetime	Date and time information.
FrameTarget	Frame name used as destination for results of certain actions.
LanguageCode	A language code, as per [RFC1766] [p.167] .
Length	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.

LinkTypes	Authors may use the following recognized link types, listed here with their conventional interpretations. A LinkTypes value refers to a space-separated list of link types. Whitespace characters are not permitted within link types. These link types are case-insensitive, i.e., "Alternate" has the same meaning as "alternate". User agents, search engines, etc. may interpret these link types in a variety of ways. For example, user agents may provide access to linked documents through a navigation bar. **Alternate** Designates substitute versions for the document in which the link occurs. When used together with the `xml:lang` attribute, it implies a translated version of the document. When used together with the `media` attribute, it implies a version designed for a different medium (or media). **Stylesheet** Refers to an external style sheet. See the Style Module [p.43] for details. This is used together with the link type "Alternate" for user-selectable alternate style sheets. **Start** Refers to the first document in a collection of documents. This link type tells search engines which document is considered by the author to be the starting point of the collection. **Next** Refers to the next document in a linear sequence of documents. User agents may choose to preload the "next" document, to reduce the perceived load time. **Prev** Refers to the previous document in an ordered series of documents. Some user agents also support the synonym "Previous". **Contents** Refers to a document serving as a table of contents. Some user agents also support the synonym *ToC* (from "Table of Contents"). **Index** Refers to a document providing an index for the current document. **Glossary** Refers to a document providing a glossary of terms that pertain to the current document. **Copyright** Refers to a copyright statement for the current document. **Chapter** Refers to a document serving as a chapter in a collection of documents. **Section** Refers to a document serving as a section in a collection of documents. **Subsection** Refers to a document serving as a subsection in a collection of documents. **Appendix** Refers to a document serving as an appendix in a collection of documents. **Help** Refers to a document offering help (more information, links to other sources information, etc.) **Bookmark** Refers to a bookmark. A bookmark is a link to a key entry point within an extended document. The title attribute may be used, for example, to label the bookmark. Note that several bookmarks may be defined in each document.

MediaDesc	The MediaDesc attribute is a comma-separated list of media descriptors. The following is a list of recognized media descriptors:
	screen
	Intended for non-paged computer screens.
	tty
	Intended for media using a fixed-pitch character grid, such as teletypes, terminals, or portable devices with limited display capabilities.
	tv
	Intended for television-type devices (low resolution, color, limited scrollability).
	projection
	Intended for projectors.
	handheld
	Intended for handheld devices (small screen, monochrome, bitmapped graphics, limited bandwidth).
	print
	Intended for paged, opaque material and for documents viewed on screen in print preview mode.
	braille
	Intended for braille tactile feedback devices.
	aural
	Intended for speech synthesizers.
	all
	Suitable for all devices.
	Future versions of XHTML may introduce new values and may allow parameterized values. To facilitate the introduction of these extensions, conforming user agents must be able to parse the media attribute value as follows:
	1. The value is a comma-separated list of entries. For example,
	`media="screen, 3d-glasses, print and resolution > 90dpi"`
	is mapped to:
	`"screen"`
	`"3d-glasses"`
	`"print and resolution > 90dpi"`
	2. Each entry is truncated just before the first character that isn't a US ASCII letter [a-zA-Z] (ISO 10646 hex 41-5a, 61-7a), digit [0-9] (hex 30-39), or hyphen (hex 2d). In the example, this gives:
	`"screen"`
	`"3d-glasses"`
	`"print"`
	3. A case-sensitive match is then made with the set of media types defined above. User agents may ignore entries that don't match. In the example we are left with `screen` and `print`.
	Note. *Style sheets may include media-dependent variations within them (e.g., the CSS **@media** construct). In such cases it may be appropriate to use "media =all".*

MultiLength	The value may be a Length or a relative length. A relative length has the form "i*", where "i" is an integer. When allotting space among elements competing for that space, user agents allot pixel and percentage lengths first, then divide up remaining available space among relative lengths. Each relative length receives a portion of the available space that is proportional to the integer preceding the "*". The value "*" is equivalent to "1*". Thus, if 60 pixels of space are available after the user agent allots pixel and percentage space, and the competing relative lengths are 1*, 2*, and 3*, the 1* will be allotted 10 pixels, the 2* will be allotted 20 pixels, and the 3* will be allotted 30 pixels.
Number	One or more digits
Pixels	The value is an integer that represents the number of pixels of the canvas (screen, paper). Thus, the value "50" means fifty pixels. For normative information about the definition of a pixel, please consult [CSS2] [p.167]
Script	Script data can be the content of the "script" element and the value of intrinsic event attributes. User agents must not evaluate script data as HTML markup but instead must pass it on as data to a script engine. The case-sensitivity of script data depends on the scripting language. Please note that script data that is element content may not contain character references, but script data that is the value of an attribute may contain them.
Text	Arbitrary textual data, likely meant to be human-readable.
URI	A Uniform Resource Identifier, as per [URI] [p.168] .
URIs	A space-separated list of Uniform Resource Identifiers, as per [URI] [p.168] .

4.4. An Example Abstract Module Definition

This section is informative

This section defines a sample abstract module as an example of how to take advantage of the syntax rules defined above. Since this example is trying to use all of the various syntactic elements defined, it is pretty complicated. Typical module definitions would be much simpler than this. Finally, note that this module references the attribute collection Common. This is a collection defined in the XHTML Modularization specification that includes all of the basic attributes that most elements need.

4.4.1. XHTML Skiing Module

The XHTML Skiing Module defines markup used when describing aspects of a ski lodge. The elements and attributes defined in this module are:

Elements	Attributes	Minimal Content Model
resort	Common, href (CDATA)	description , Aspen+
lodge	Common	description, (Aspen - lift)+
lift	Common, href	description?
chalet	Common, href	description?
room	Common, href	description?
lobby	Common, href	description?
fireplace	Common, href	description?
description	Common	PCDATA*

This module also defines the content set Aspen with the minimal content model lodge | lift | chalet | room | lobby | fireplace.

5. XHTML Abstract Modules

This section is *normative*.

This section specifies the contents of the XHTML abstract modules. These modules are abstract definitions of collections of elements, attributes, and their content models. These abstract modules can be mapped onto any appropriate specification mechanism. XHTML DTD Module Implementations [p.75] , for example, maps these modules onto DTDs as described in [XML] [p.168] .

Content developers and device designers should view this section as a guide to the definition of the functionality provided by the various XHTML-defined modules. When developing documents or defining a profile for a class of documents, content developers can determine which of these modules are essential for conveying their message. When designing clients, device designers should develop their device profiles by choosing from among the abstract modules defined here.

Except when overridden in this document, the semantics of these elements and attributes are defined in [HTML4] [p.167] .

5.1. Attribute Collections

Many of the abstract modules in this section define the required attributes for elements. The table below defines some collections of attributes that are referenced throughout the modules. These expressions should in no way be considered normative or mandatory. They are an editorial convenience for this document. When used in the remainder of this section, it is the expansion of the term that is normative, not the term itself.

The following basic attribute sets are used on many elements. In each case where they are used, their use is identified via their name rather than enumerating the list.

Collection Name	Attributes in Collection
Core	class (NMTOKENS [p.20]), id (ID [p.20]), title (CDATA [p.20])
I18N	xml:lang (NMTOKEN [p.20])
Events	onclick (Script [p.24]), ondblclick (Script [p.24]), onmousedown (Script [p.24]), onmouseup (Script [p.24]), onmouseover (Script [p.24]), onmousemove (Script [p.24]), onmouseout (Script [p.24]), onkeypress (Script [p.24]), onkeydown (Script [p.24]), onkeyup (Script [p.24])
Style	style (CDATA [p.20])
Common	Core [p.27] + Events [p.27] + I18N [p.27] + Style [p.27]

Note that the Events collection is only defined when the Intrinsic Events Module is selected. Otherwise, the Events collection is empty.

Also note that the Style collection is only defined when the Style Attribute Module is selected. Otherwise, the Style collection is empty.

5.2. Core Modules

The core modules are modules that are required to be present in any XHTML Family Conforming Document Type [p.15] .

5.2.1. Structure Module

The Structure 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	Minimal Content Model
body	Common [p.27]	(Heading | Block | List)*
head	I18N [p.27] , profile (URI [p.24])	title
html	I18N [p.27] , version (CDATA [p.20]), xmlns (URI [p.24] = "http://www.w3.org/1999/xhtml")	head, body
title	I18N [p.27]	PCDATA

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 attribute is defined to be "http://www.w3.org/1999/xhtml". Also note that because the xmlns attribute is treated specially by XML Namespace-aware parsers [XMLNAMES [p.168]], it is legal to have it present as an attribute of each element. However, any time the xmlns attribute is used in the context of an XHTML module, whether with a prefix or not, the value of the attribute shall be the XHTML namespace defined here. See Defining the Namespace of a Module [p.56] for more on rules regarding namespace usage with XHTML family modules.

Implementation: DTD [p.97]

5.2.2. Text Module

This module defines all of the basic text container elements, attributes, and their content model:

Element	Attributes	Minimal Content Model
abbr	Common [p.27]	(PCDATA | Inline)*
acronym	Common [p.27]	(PCDATA | Inline)*
address	Common [p.27]	(PCDATA | Inline)*
blockquote	Common [p.27] , cite (URI [p.24])	(PCDATA | Heading | Block)*
br	Core [p.27]	EMPTY
cite	Common [p.27]	(PCDATA | Inline)*
code	Common [p.27]	(PCDATA | Inline)*
dfn	Common [p.27]	(PCDATA | Inline)*
div	Common [p.27]	(Heading | Block | List)*
em	Common [p.27]	(PCDATA | Inline)*
h1	Common [p.27]	(PCDATA | Inline)*
h2	Common [p.27]	(PCDATA | Inline)*
h3	Common [p.27]	(PCDATA | Inline)*
h4	Common [p.27]	(PCDATA | Inline)*
h5	Common [p.27]	(PCDATA | Inline)*
h6	Common [p.27]	(PCDATA | Inline)*
kbd	Common [p.27]	(PCDATA | Inline)*
p	Common [p.27]	(PCDATA | Inline)*
pre	Common [p.27] , xml:space="preserve"	(PCDATA | Inline)*
q	Common [p.27] , cite (URI [p.24])	(PCDATA | Inline)*
samp	Common [p.27]	(PCDATA | Inline)*
span	Common [p.27]	(PCDATA | Inline)*
strong	Common [p.27]	(PCDATA | Inline)*
var	Common [p.27]	(PCDATA | Inline)*

The minimal content model for this module defines some content sets:

Heading
  h1 | h2 | h3 | h4 | h5 | h6
Block
  address | blockquote | div | p | pre
Inline
  abbr | acronym | br | cite | code | dfn | em | kbd | q | samp | span | strong | var
Flow
  Heading | Block | Inline

Implementation: DTD [p.99]

5.2.3. Hypertext Module

The Hypertext Module provides the element that is used to define hypertext links to other resources. This module supports the following element and attributes:

Element	Attributes	Minimal Content Model
a	Common [p.27] , accesskey (Character [p.21]), charset (Charset [p.21]), href (URI [p.24]), hreflang (LanguageCode [p.21]), rel (LinkTypes [p.22]), rev (LinkTypes [p.22]), tabindex (Number [p.24]), type (ContentType [p.21])	(PCDATA | Inline - a)*

This module adds the a element to the Inline content set of the Text Module.

Implementation: DTD [p.100]

5.2.4. List Module

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

Elements	Attributes	Minimal Content Model
dl	Common [p.27]	(dt | dd)+
dt	Common [p.27]	(PCDATA | Inline)*
dd	Common [p.27]	(PCDATA | Inline)*
ol	Common [p.27]	li+
ul	Common [p.27]	li+
li	Common [p.27]	(PCDATA | Inline)*

This module also defines the content set List with the minimal content model (dl | ol | ul)+ and adds this set to the Flow content set of the Text Module.

Implementation: DTD [p.101]

5.3. Applet Module

This module is deprecated. Similar functionality can be found in the Object Module [p.39] .

The Applet Module provides elements for referencing external applications. Specifically, the Applet Module supports the following elements and attributes:

Element	Attributes	Minimal Content Model
applet	Core, alt* (Text [p.24]), archive (CDATA [p.20]), code (CDATA [p.20]), codebase (URI [p.24]), height* (Length [p.21]), object (CDATA [p.20]), width* (Length [p.21])	param?
param	id (ID [p.20]), name* (CDATA [p.20]), type (ContentType [p.21]), value (CDATA [p.20]), valuetype ("data"* | "ref" | "object")	EMPTY

When the Applet Module is used, it adds the `applet` element to the Inline content set of the Text Module.

Implementation: DTD [p.103]

5.4. Text Extension Modules

This section defines a variety of additional textual markup modules.

5.4.1. Presentation Module

This module defines elements, attributes, and a minimal content model for simple presentation-related markup:

Element	Attributes	Minimal Content Model
b	Common [p.27]	(PCDATA | Inline)*
big	Common [p.27]	(PCDATA | Inline)*
hr	Common [p.27]	EMPTY
i	Common [p.27]	(PCDATA | Inline)*
small	Common [p.27]	(PCDATA | Inline)*
sub	Common [p.27]	(PCDATA | Inline)*
sup	Common [p.27]	(PCDATA | Inline)*
tt	Common [p.27]	(PCDATA | Inline)*

When this module is used, the `hr` element is added to the Block content set of the Text Module. In addition, the `b, big, i, small, sub, sup,` and `tt` elements are added to the Inline content set of the Text Module.

Implementation: DTD [p.104]

5.4.2. Edit Module

This module defines elements and attributes for use in editing-related markup:

Element	Attributes	Minimal Content Model
del	Common [p.27] , cite (URI [p.24]), datetime (Datetime [p.21])	(PCDATA | Inline)*
ins	Common [p.27] , cite (URI [p.24]), datetime (Datetime [p.21])	(PCDATA | Inline)*

When this module is used, the `del` and `ins` elements are added to the Inline content set of the Text Module.

Implementation: DTD [p.105]

5.4.3. Bi-directional Text Module

The Bi-directional Text module defines an element that can be used to declare the bi-directional rules for the element’s content.

Elements	Attributes	Minimal Content Model
bdo	Core [p.27] , dir* ("ltr" | "rtl")	(PCDATA | Inline)*

When this module is used, the `bdo` element is added to the Inline content set of the Text Module. Selecting this module also adds the attribute `dir* ("ltr" | "rtl")` to the I18N attribute collection.

Implementation: DTD [p.106]

5.5. Forms Modules

5.5.1. Basic Forms Module

The Basic Forms Module provides the form-related elements, but only in a limited form. Specifically, the Basic Forms Module supports the following elements, attributes, and minimal content model:

Elements	Attributes	Minimal Content Model
form	Common [p.27] , action* (URI [p.24]), method ("get"* | "post"), enctype (ContentType [p.21])	Heading | Block - form
input	Common [p.27] , accesskey (Character [p.21]), checked ("checked"), maxlength (Number [p.24]), name (CDATA [p.20]), size (Number [p.24]), src (URI [p.24]), type ("text"* | "password" | "checkbox" | "radio" | "submit" | "reset" | "hidden"), value (CDATA [p.20])	EMPTY
label	Common [p.27] , accesskey (Character [p.21]), for (IDREF [p.20])	(PCDATA | Inline - label)*
select	Common [p.27] , multiple ("multiple"), name (CDATA [p.20]), size (Number [p.24])	option+
option	Common [p.27] , selected ("selected"), value (CDATA [p.20])	Inline*
textarea	Common [p.27] , accesskey (Character [p.21]), cols* (Number [p.24]), name (CDATA [p.20]), rows* (Number [p.24])	PCDATA*

This module defines two content sets:

Form
  form
Formctrl
  input | label | select | textarea

When this module is used, it adds the Form content set to the Block content set and it adds the Formctrl content set to the Inline content set as these are defined in the Text Module.

Implementation: DTD [p.107]

5.5.2. Forms Module

The Forms Module provides all of the forms features found in HTML 4.0. Specifically, the Forms Module supports:

Elements	Attributes	Minimal Content Model
form	Common [p.27] , accept (ContentTypes [p.21]), accept-charset (Charsets [p.21]), action* (URI [p.24]), method ("get"* | "post"), enctype (ContentType [p.21])	(Heading | Block - form | fieldset)+
input	Common [p.27] , accept (ContentTypes [p.21]), accesskey (Character [p.21]), alt (CDATA [p.20]), checked ("checked"), disabled ("disabled"), maxlength (Number [p.24]), name (CDATA [p.20]), readonly ("readonly"), size (Number [p.24]), src (URI [p.24]), tabindex (Number [p.24]), type ("text"* | "password" | "checkbox" | "button" | "radio" | "submit" | "reset" | "file" | "hidden" | "image"), value (CDATA [p.20])	EMPTY
select	Common [p.27] , disabled ("disabled"), multiple ("multiple"), name (CDATA [p.20]), size (Number [p.24]), tabindex (Number [p.24])	(optgroup | option)+
option	Common [p.27] , disabled ("disabled"), label (Text [p.24]), selected ("selected"), value (CDATA [p.20])	PCDATA
textarea	Common [p.27] , accesskey (Character [p.21]), cols* (Number [p.24]), disabled ("disabled"), name (CDATA [p.20]), readonly ("readonly"), rows* (Number [p.24]), tabindex (Number [p.24])	PCDATA
button	Common [p.27] , accesskey (Character [p.21]), disabled ("disabled"), name (CDATA [p.20]), tabindex (Number [p.24]), type ("button" | "submit"* | "reset"), value (CDATA [p.20])	(PCDATA | Heading | List | Block - Form | Inline - Formctrl)*
fieldset	Common [p.27]	(PCDATA | legend | Flow)*
label	Common [p.27] , accesskey (Character [p.21]), for (IDREF [p.20])	(PCDATA | Inline - label)*
legend	Common [p.27] , accesskey (Character [p.21])	(PCDATA | Inline)+
optgroup	Common [p.27] , disabled ("disabled"), label* (Text [p.24])	option+

This module defines two content sets:

Form
    form | fieldset
Formctrl
    input | select | textarea | label | button

When this module is used, it adds the Form content set to the Block content set and it adds the Formctrl content set to the Inline content set as these are defined in the Text Module.

The Forms Module is a superset of the Basic Forms Module. These modules may not be used together in a single document type.

Implementation: DTD [p.110]

5.6. Table Modules

5.6.1. Basic Tables Module

The Basic Tables Module provides table-related elements, but only in a limited form. Specifically, the Basic Tables Module supports:

Elements	**Attributes**	**Minimal Content Model**
caption	Common [p.27]	(PCDATA | Inline)*
table	Common [p.27] , summary (Text [p.24]), width (Length [p.21])	caption?, tr+
td	Common [p.27] , abbr (Text [p.24]), align ("left" | "center" | "right"), axis (CDATA [p.20]), colspan (Number [p.24]), headers (IDREFS [p.20]), rowspan (Number [p.24]), scope ("row" | "col"), valign ("top" | "middle" | "bottom")	(PCDATA | Flow - table)*
th	Common [p.27] , abbr (Text [p.24]), align ("left" | "center" | "right"), axis (CDATA [p.20]), colspan (Number [p.24]), headers (IDREFS [p.20]), rowspan (Number [p.24]), scope ("row" | "col" | "rowgroup" | "colgroup"), valign ("top" | "middle" | "bottom")	(PCDATA | Flow - table)*
tr	Common [p.27] , align ("left" | "center" | "right"), valign ("top" | "middle" | "bottom")	(td)+

When this module is used, it adds the `table` element to the Block content set as defined in the Text Module.

Implementation: DTD [p.116]

5.6.2. Tables Module

As its name suggests, the Tables Module provides table-related elements that are better able to be accessed by non-visual user agents. Specifically, the Tables Module supports the following elements, attributes, and content model:

Elements	Attributes	Minimal Content Model
caption	Common [p.27]	(PCDATA | Inline)*
table	Common [p.27] , border (Pixels [p.24]), cellpadding (Length [p.21]), cellspacing (Length [p.21]), datapagesize (CDATA [p.20]), frame ("void" | "above" | below" | "hsides" | "lhs" | "rhs" | "vsides" | "box" | "border"), rules ("none" | "groups" | "rows" | "cols" | "all"), summary (Text [p.24]), width (Length [p.21])	caption?, (col* | colgroup*), ((thead?, tfoot?, tbody+) | (tr+))
td	Common [p.27] , abbr (Text [p.24]), align ("left" | "center" | "right" | "justify" | "char"), axis (CDATA [p.20]), char (Character [p.21]), charoff (Length [p.21]), colspan (Number [p.24]), headers (IDREFS [p.20]), rowspan (Number [p.24]), scope ("row", "col", "rowgroup", "colgroup"), valign ("top" | "middle" | "bottom" | "baseline")	(PCDATA | Flow)*
th	Common [p.27] , abbr (Text [p.24]), align ("left" | "center" | "right" | "justify" | "char"), axis (CDATA [p.20]), char (Character [p.21]), charoff (Length [p.21]), colspan (Number [p.24]), headers (IDREFS [p.20]), rowspan (Number [p.24]), scope ("row", "col", "rowgroup", "colgroup"), valign ("top" | "middle" | "bottom" | "baseline")	(PCDATA | Flow)*
tr	Common [p.27] , align ("left" | "center" | "right" | "justify", "char"), char (Character [p.21]), charoff (Length [p.21]), valign ("top" | "middle" | "bottom" | "baseline")	(td | th)+
col	Common [p.27] , align ("left" | "center" | "right" | "justify", "char"), char (Character [p.21]), charoff (Length [p.21]), span (Number [p.24]), valign ("top" | "middle" | "bottom" | "baseline"), width (MultiLength [p.24])	EMPTY
colgroup	Common [p.27] , align ("left" | "center" | "right" | "justify", "char"), char (Character [p.21]), charoff (Length [p.21]), span (Number [p.24]), valign ("top" | "middle" | "bottom" | "baseline"), width (MultiLength [p.24])	col*
tbody	Common [p.27] , align ("left" | "center" | "right" | "justify", "char"), char (Character [p.21]), charoff (Length [p.21]), valign ("top" | "middle" | "bottom" | "baseline")	tr+
thead	Common [p.27] , align ("left" | "center" | "right" | "justify", "char"), char (Character [p.21]), charoff (Length [p.21]), valign ("top" | "middle" | "bottom" | "baseline")	tr+

Elements	Attributes	Minimal Content Model
tfoot	Common [p.27] , align ("left" | "center" | "right" | "justify", "char"), char (Character [p.21]), charoff (Length [p.21]), valign ("top" | "middle" | "bottom" | "baseline")	tr+

When this module is used, it adds the `table` element to the Block content set of the Text Module.

Implementation: DTD [p.119]

5.7. Image Module

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	Minimal Content Model
img	Common [p.27] , alt* (Text [p.24]), height (Length [p.21]), longdesc (URI [p.24]), src* (URI [p.24]), width (Length [p.21])	EMPTY

When this module is used, it adds the `img` element to the Inline content set of the Text Module.

Implementation: DTD [p.125]

5.8. Client-side Image Map Module

The Client-side Image Map Module provides elements for client side image maps. It requires that the Image Module (or another module that supports the `img` element) be included. The Client-side Image Map Module supports the following elements:

Elements	Attributes	Minimal Content Model
a&	coords (CDATA [p.20]), shape ("rect" | "circle" | "poly" | "default")	n/a
area	Common [p.27] , accesskey (Character [p.21]), alt* (Text [p.24]), coords (CDATA [p.20]), href (URI [p.24]), nohref ("nohref"), shape ("rect"* | "circle" | "poly" | "default"), tabindex (Number [p.24])	EMPTY
img&	usemap (IDREF [p.20])	n/a
map	I18N [p.27] , Events [p.27] , class (NMTOKEN [p.20]), id* (ID [p.20]), title (CDATA [p.20])	((Heading | Block) | area)+
object&	usemap (IDREF [p.20])	Note: Only when the object module is included

When this module is used, the `map` element is added to the Inline content set of the Text Module.

Implementation: DTD [p.126]

5.9. Server-side Image Map Module

The Server-side Image Map Module provides support for image-selection and transmission of selection coordinates. It requires that the Image Module (or another module that supports the `img` element) be included. The Server-side Image Map Module supports the following attributes:

Elements	Attributes	Minimal Content Model
img&	ismap ("ismap")	n/a

Implementation: DTD [p.128]

5.10. Object Module

The Object Module provides elements for general-purpose object inclusion. Specifically, the Object Module supports:

Elements	Attributes	Minimal Content Model
object	Common, archive (URIs [p.24]), classid (URI [p.24]), codebase (URI [p.24]), codetype (ContentType [p.21]), data (URI [p.24]), declare ("declare"), height (Length [p.21]), standby (Text [p.24]), tabindex (Number [p.24]), type (ContentType [p.21]), width (Length [p.21])	(PCDATA | Flow | param)*
param	id (ID [p.20]), name* (CDATA [p.20]), type (ContentType [p.21]), value (CDATA [p.20]), valuetype ("data"* | "ref" | "object")	EMPTY

When this module is used, it adds the `object` element to the Inline content set of the Text Module.

Implementation: DTD [p.128]

5.11. Frames Module

As its name suggests, the Frames Module provides frame-related elements. Specifically, the Frames Module supports:

Elements	Attributes	Minimal Content Model
frameset	Core [p.27] , cols (MultiLength [p.24]), rows (MultiLength [p.24])	frame+, noframes?
frame	Core [p.27] , frameborder ("1" | "0"), longdesc (URI [p.24]), marginheight (Pixels [p.24]), marginwidth (Pixels [p.24]), noresize ("noresize"), scrolling ("yes" | "no" | "auto"*), src (URI [p.24])	EMPTY
noframes	Common [p.27]	body

When this module is selected, the minimal content model of the `html` element of the structure module is changed to `(head, frameset)`.

Implementation: DTD [p.129]

5.12. Target Module

The content of a frame can specify destination targets for a selection. This module adds the `target` element to the area and link defining elements. This is definied as a separate module so it can be included in documents that will be included in frames and documents that use the `target` feature to open a new window.

Elements	**Attributes**	**Notes**
a&	target (CDATA [p.21])	
area&	target (CDATA [p.21])	When the Client-side Image Map module is selected.
base&	target (CDATA [p.21])	When the Legacy module is selected.
link&	target (CDATA [p.21])	When the Link Module is selected.
form&	target (CDATA [p.21])	When the Basic Forms or Forms module is selected.

Implementation: DTD [p.131]

5.13. Iframe Module

The Iframe Module defines an element for the definition of inline frames. The element and attribute included in this module are:

Elements	**Attributes**	**Minimal Content Model**
iframe	Core [p.27] , frameborder ("1" | "0"), height (Pixels [p.24]), longdesc (URI [p.24]), marginheight (Pixels [p.24]), marginwidth (Pixels [p.24]), scrolling ("yes" | "no" | "auto"*), src (URI [p.24]), width (Length [p.21])	Flow

When this module is used, the `iframe` element is added to the Inline content set as defined by the Text Module.

Implementation: DTD [p.132]

5.14. Intrinsic Events Module

Intrinsic events are attributes that are used in conjunction with elements that can have specific actions occur when certain events are performed by the user. The attributes indicated in the following table are added to the attribute set for their respective elements *only* when the modules defining those elements are selected. Note also that selection of this module defines the attribute collection Events [p.27] as described above. Attributes defined by this module are:

Elements	Attributes	Notes
a&	onblur (Script [p.24]), onfocus (Script [p.24])	
area&	onblur (Script [p.24]), onfocus (Script [p.24])	When the Client-side Image Map module is also used
form&	onreset (Script [p.24]), onsubmit (Script [p.24])	When the Basic Forms or Forms module is used
body&	onload (Script [p.24]), onunload (Script [p.24])	
label&	onblur (Script [p.24]), onfocus (Script [p.24])	When the Forms module is used
input&	onblur (Script [p.24]), onchange (Script [p.24]), onfocus (Script [p.24]), onselect (Script [p.24])	When the Basic Forms or Forms module is used
select&	onblur (Script [p.24]), onchange (Script [p.24]), onfocus (Script [p.24])	When the Basic Forms or Forms module is used
textarea&	onblur (Script [p.24]), onchange (Script [p.24]), onfocus (Script [p.24]), onselect (Script [p.24])	When the Basic Forms or Forms module is used
button&	onblur (Script [p.24]), onfocus (Script [p.24])	When the Forms module is used

Implementation: DTD [p.133]

5.15. Metainformation Module

The Metainformation Module defines an element that describes information within the declarative portion of a document (in XHTML within the head element). This module includes the following element:

Elements	Attributes	Minimal Content Model
meta	I18N [p.27] , content* (CDATA [p.20]), http-equiv (NMTOKEN [p.20]), name (NMTOKEN [p.20]), scheme (CDATA [p.20])	EMPTY

When this module is selected, the `meta` element is added to the content model of the `head` element as defined in the Structure Module.

Implementation: DTD [p.135]

5.16. Scripting Module

The Scripting Module defines elements that are used to contain information pertaining to executable scripts or the lack of support for executable scripts. Elements and attributes included in this module are:

Elements	Attributes	Minimal Content Model
noscript	Common [p.27]	(Heading | List | Block)+
script	charset (Charset [p.21]), defer ("defer"), src (URI [p.24]), type* (ContentType [p.21]), xml:space="preserve"	PCDATA

When this module is used, the `script` and `noscript` elements are added to the Block and Inline content sets of the Text Module. In addition, the `script` element is added to the content model of the `head` element defined in the Structure Module.

Implementation: DTD [p.136]

5.17. Stylesheet Module

The Stylesheet Module defines an element to be used when declaring internal stylesheets. The element and attributes defined by this module are:

Elements	Attributes	Minimal Content Model
style	I18N [p.27] , media (MediaDesc [p.23]), title (Text [p.24]), type* (ContentType [p.21]), xml:space="preserve"	PCDATA

When this module is used, it adds the `style` element to the content model of the `head` element of the Structure Module.

Implementation: DTD [p.137]

5.18. Style Attribute Module

The Style Attribute Module defines the `style` attribute. When this module is selected, it activates the Style Attribute Collection [p.27] .

Implementation: DTD [p.138]

5.19. Link Module

The Link Module defines an element that can be used to define links to external resources. These resources are often used to augment the user agent's ability to process the associated XHTML document. The element and attributes included in this module are:

Elements	**Attributes**	**Minimal Content Model**
link	Common [p.27] , charset (Charset [p.21]), href (URI [p.24]), hreflang (LanguageCode [p.21]), media (MediaDesc [p.23]), rel (LinkTypes [p.22]), rev (LinkTypes [p.22]), type (ContentType [p.21])	EMPTY

When this module is used, it adds the `link` element to the content model of the `head` element as defined in the Structure Module.

Implementation: DTD [p.139]

5.20. Base Module

The Base Module defines an element that can be used to define a base URI against which relative URIs in the document will be resolved. The element and attribute included in this module are:

Elements	**Attributes**	**Minimal Content Model**
base	href* (URI [p.24])	EMPTY

When this module is used, it adds the `base` element to the content model of the `head` element of the Structure Module.

Implementation: DTD [p.140]

5.21. Name Identification Module

This module is deprecated.

The Name Identification module defines the attribute `name` for a collection of elements. The `name` attribute was used historically to identify certain elements within HTML documents. While the `name` attribute has been supplanted by the `id` attribute in all of these elements, there may be instances where markup languages will wish to support both. Such markup languages may do so by including this module.

Note that by including this module, both the `name` and `id` attributes are defined for the elements indicated. In this situation, if the `name` attribute is defined for an element, the `id` attribute must also be defined. Further, these attributes must both have the same value. Finally, when documents that use this attribute are served as Internet Media Type "text/xml" or "application/xml", the value of the `name` attribute on these elements shall not be used as a fragment identifier.

Elements	Attributes	Notes
a&	name (CDATA [p.20])	
applet&	name (CDATA [p.20])	When the Applet module is selected.
form&	name (CDATA [p.20])	When the Forms or Basic Forms module is selected.
frame&	name (CDATA [p.20])	When the Frames module is selected.
iframe&	name (CDATA [p.20])	When the Iframe module is selected.
img&	name (CDATA [p.20])	When the Image Module is selected.
map&	name (CDATA [p.20])	When the Client-side Image Map module is selected.

Implementation: DTD [p.141]

5.22. Legacy Module

The Legacy Module defines elements and attributes that were deprecated in previous versions of HTML and XHTML. While the use of these elements and attributes is no longer encouraged, they are provided in this module to ease their integration should markup language authors wish to support them.

The following table defines the elements and attributes that are defined when the Legacy module is selected.

Elements	Attributes	Minimal Content Model
basefont	color (Color [p.21]), face (CDATA [p.20]), id (ID [p.20]), size (CDATA [p.20])	EMPTY
center	Common [p.27]	(PCDATA | Flow)*
font	Common [p.27] , color (Color [p.21]), face (CDATA [p.20]), size (CDATA [p.20])	(PCDATA | Inline)*
s	Common [p.27]	(PCDATA | Inline)*
strike	Common [p.27]	(PCDATA | Inline)*
u	Common [p.27]	(PCDATA | Inline)*

The following table shows additional attributes for elements defined elsewhere when the Legacy module is selected.

Elements	Attributes	Notes
body&	alink (Color [p.21]), background (URI [p.24]), bgcolor (Color [p.21]), link (Color [p.21]), text (Color [p.21]), vlink (Color [p.21])	
br&	clear ("left" | "all" | "right" | "none"*)	
caption&	align ("left" | "center" | "right" | "justify")	
div&	align ("left" | "center" | "right" | "justify")	
h1-h6&	align ("left" | "center" | "right" | "justify")	
hr&	align ("left" | "center" | "right" | "justify"), noshade ("noshade"), size (Pixels [p.24]), width (Length [p.21]),	
img&	align ("left" | "center" | "right" | "justify"), border (Pixels [p.24]), hspace (Pixels [p.24]), vspace (Pixels [p.24])	
input&	align ("left" | "center" | "right" | "justify")	When the Basic Forms or Forms module is selected.
legend&	align ("left" | "center" | "right" | "justify")	When the Forms module is selected.
li&	type (CDATA [p.20]), value (Number [p.24])	
ol&	compact ("compact"), start (Number [p.24]), type (CDATA [p.20])	

Elements	Attributes	Notes
p&	align ("left" | "center" | "right", "justify")	
pre&	width (Number [p.24])	
script&	language (CDATA [p.20])	When the Scripting module is selected.
table&	align ("left" | "center" | "right" | "justify"), bgcolor (Color [p.21])	When the Tables module is selected.
tr&	bgcolor (Color [p.21])	When the Tables module is selected.
th&	bgcolor (Color [p.21]), height (Pixels [p.24]) nowrap ("nowrap"), width (Pixels [p.24])	When the Tables module is selected.
td&	bgcolor (Color [p.21]), height (Pixels [p.24]) nowrap ("nowrap"), width (Pixels [p.24])	When the Tables module is selected.
ul&	compact ("compact"), type (CDATA [p.20])	

Implementation: DTD [p.142]

A. Building Schema Modules

This appendix is *normative*.

This appendix will contain instructions on how to define XML Schema modules that are compatible with the XHTML Modularization implementation via XML Schema [XMLSCHEMA] [p.168] when the XML Schema becomes a W3C approved recommendation.

B. Developing Schema with defined and extended modules

This appendix is *normative*.

This appendix will contain instructions on how to define entire markup languages using XHTML modules via XML Schema [XMLSCHEMA] [p.168] when the XML Schema becomes a W3C approved recommendation.

C. XHTML Schema Module Implementations

This appendix is *normative*.

This appendix will contain implementations of the modules defined in XHTML Abstract Modules [p.27] via XML Schema [XMLSCHEMA] [p.168] when the XML Schema becomes a W3C approved recommendation.

D. Building DTD Modules

This section is *normative.*

XHTML modules are implemented as DTD fragments. When these fragments are assembled in a specific manner (described in Developing DTDs with defined and extended modules [p.63]), the resulting DTD is a representation of a complete document type. This representation can then be used for validation of instances of the document type.

The key to combining these fragments into a meaningful DTD is the rules used to define the fragments. This section defines those rules. When these rules are followed, DTD authors can be confident that their modules will interface cleanly with other XHTML-compatible modules.

Modules conforming to these rules also need to satisfy the conformance requirements defined in XHTML Family Module Conformance [p.16] in order to be called *XHTML Family Modules*.

D.1. Parameter Entity Naming

This specification classifies parameter entities into seven categories and names them consistently using the following suffixes:

.mod
: parameter entities use the suffix `.mod` when they are used to represent a DTD module (a collection of elements, attributes, parameter entities, etc). In this specification, each module is an atomic unit and may be represented as a separate file entity.

.module
: parameter entities use the suffix `.module` when they are used to control the inclusion of a DTD module by containing either of the conditional section keywords `INCLUDE` or `IGNORE`.

.qname
: parameter entities use the suffix `.qname` when they are used to represent the qualified name of an element. See Defining the Namespace of a Module [p.56] for more information on qualified names.

.content
: parameter entities use the suffix `.content` when they are used to represent the content model of an element type.

.class
: parameter entities use the suffix `.class` when they are used to represent elements of the same class.

.mix
: parameter entities use the suffix `.mix` when they are used to represent a collection of element types from different classes.

.attrib
: parameter entities use the suffix `.attrib` when they are used to represent a group of tokens representing one or more complete attribute specifications within an ATTLIST declaration.

For example, in HTML 4, the `%block;` parameter entity is defined to represent the heterogeneous collection of element types that are block-level elements. In this specification, the corollary parameter entity is `%Block.mix;`.

When defining parameter entities in the classes defined here, modules should scope the names of the entities by using unique prefixes. For example, the content model for the element `myelement` in the module mymodule could be named `MYMODULE.myelement.content`. Other schemes are possible. Regardless of the scheme used, module authors should strive to ensure that parameter entities they define are named uniquely so that they do not collide with other parameter entities and so that the interface methods for the module are obvious to its users.

D.2. Defining the Namespace of a Module

XHTML requires that the elements and attributes declared in a module be within a defined XML Namespace [XMLNAMES] [p.168] . The identification of this namespace is an arbitrary URI. XHTML requires that when a module is implemented using an XML DTD, the module declares the namespace in a special manner. The purpose of this is to permit the selection, at document parse/validation time, of the use of namespace prefixes and of the *prefix* that is used to identify elements and attributes from the module.

Content developers who wish to develop documents based upon hybrid document types may choose to use XML Namespace prefixes on elements from the XHTML namespace, on elements from other namespaces, or on both. In order to ensure that such documents are XHTML conforming and backward compatible with non-namespace aware tools, the W3C recommends that content developers *do not* use XML Namespace prefixes on elements from the XHTML namespace. When content developers are interested in having their content processed by namespace-aware processors, the W3C further recommends that elements in non-XHTML namespaces be specified using an XML Namespace prefix rather than relying upon XML Namespace defaulting mechanisms.

Each XHTML-conforming module implemented as an XML DTD is required to define a default XML Namespace prefix, a method for changing this prefix within a document instance, and a marked section that turns on the processing of the prefix.

Note that it is legal and expected for multiple modules to be part of the same namespace when they are related. All of the XHTML modules, for example, are part of the same namespace.

D.2.1. Qualified Names sub-module

First, you need to define a qualified names sub-module (a sub-module is just a *file entity* that is separated so that it can be incorporated into the ultimate DTD at the appropriate point). The qualified names sub-module is build using the following steps (where the string MODULE is replaced with an appropriate string for the new module):

1. Define a parameter entity MODULE.prefixed that announces whether the elements in the module are being used with XML Namespace prefixed names or not. This parameter entity's default value should be "%NS.prefixed;". The NS.prefixed parameter entity is defined by the XHTML framework to be IGNORE by default, and can be used in a document instance to switch on prefixing for all included namespaces.
2. Define a parameter entity MODULE.xmlns that contains the namespace identifier for this module.
3. Define a parameter entity MODULE.prefix that contains the default prefix string to use when prefixing is enabled.
4. Define a parameter entity MODULE.pfx that is "%MODULE.prefix;:" when prefixing is enabled, and "" when it is not.
5. Define a parameter entity MODULE.xmlns.extra.attrib that contains the declaration of any XML Namespace attributes for namespaces referenced by this module (e.g., xmlns:xlink). When %MODULE.prefixed is set to INCLUDE, this attribute should include the xmlns:%MODULE.pfx; declaration as well.
6. For each of the elements defined by the module, create a parameter entity of the form "MODULE.NAME.qname" to hold its qualified name. The value for this parameter entity must be "%MODULE.pfx;NAME". In this way, the parsed value will be "PREFIX:NAME" when prefixes are enabled, and "NAME" otherwise.

   If the module adds attributes to elements defined in modules that do not share the namespace of this module, declare those attributes so that they use the %MODULE.pfx prefix. For example:

   ```
   <ENTITY % MODULE.img.myattr.qname "%MODULE.pfx;myattr" >
   ```

An example of a qname sub-module for a hypothetical Inventory Module is included below:

```
<!-- ...................................................................... -->
<!-- Inventory Qname Module ................................................ -->
<!-- file: inventory-qname-1.mod

     PUBLIC "-//MY COMPANY//ELEMENTS XHTML Inventory Qnames 1.0//EN"
     SYSTEM "http://www.my.org/DTDs/inventory-qname-1.mod"

     xmlns:inventory="http://www.my.org/xmlns/inventory"
     ...................................................................... -->

<!-- Declare the default value for prefixing of this module's elements -->
<!-- Note that the NS.prefixed will get overridden by the XHTML Framework or
     by a document instance. -->
<!ENTITY % NS.prefixed "IGNORE" >
<!ENTITY % Inventory.prefixed "%NS.prefixed;" >

<!-- Declare the actual namespace of this module -->
<!ENTITY % Inventory.xmlns "http://www.my.org/xmlns/inventory" >

<!-- Declare the default prefix for this module -->
<!ENTITY % Inventory.prefix "inventory" >

<!-- Declare the prefix and any prefixed namespaces that are required by
```

```
     this module -->
<![%Inventory.prefixed;[
<!ENTITY % Inventory.pfx "%Inventory.prefix;:" >
<!ENTITY % Inventory.xmlns.extra.attrib
    "xmlns:%Inventory.prefix;   %URI.datatype;  #FIXED  ’%Inventory.xmlns;’" >
]]>
<!ENTITY % Inventory.pfx "" >
<!ENTITY % Inventory.xmlns.extra.attrib "" >

<!ENTITY % Inventory.shelf.qname "%Inventory.pfx;shelf" >
<!ENTITY % Inventory.item.qname "%Inventory.pfx;item" >
<!ENTITY % Inventory.desc.qname "%Inventory.pfx;desc" >
<!ENTITY % Inventory.sku.qname "%Inventory.pfx;sku" >
<!ENTITY % Inventory.price.qname "%Inventory.pfx;price" >
```

D.2.2. Declaration sub-module(s)

Next, you need to define one or more "declaration sub-modules". The purpose of these *file entities* is to declare the actual XML DTD Elements and Attribute lists. An XHTML declaration Module should be constructed using the following process:

1. Define a parameter entity to use within the ATTLIST of each declared element. This parameter entity should contain %NS.attrib; when %MODULE.prefixed; is set to INCLUDE, and %NS.attrib; plus "xmlns %URI.datatype; #FIXED ’%MODULE.xmlns;’" when %MODULE.prefixed; is set to IGNORE.
2. Declare all of the elements and attributes for the module. Within each ATTLIST for an element, include the parameter entity defined above so that all of the required xmlns attributes are available on each element in the module.
3. If the module adds attributes to elements defined in modules that do not share the namespace of this module, declare those attributes so that they use the %MODULE.pfx prefix. For example:

   ```
   <ENTITY % MODULE.img.myattr.qname "%MODULE.pfx;myattr" >
   <!ATTLIST %img.qname;
         "%MODULE.img.myattr.qname;    CDATA          #IMPLIED
   >
   ```

   This would add an attribute to the `img` element of the Image Module, but the attribute’s name will be the qualified name, including prefix, when prefixes are selected for a document instance.

The following example shows a declaration sub-module for a hypothetical Inventory module.

```
<!-- ...................................................................... -->
<!-- Inventory Elements Module ................................................... -->
<!-- file: inventory-1.mod

     PUBLIC "-//MY COMPANY//ELEMENTS XHTML Inventory Elements 1.0//EN"
     SYSTEM "http://www.my.org/DTDs/inventory-1.mod"

     xmlns:inventory="http://www.my.org/xmlns/inventory"
     ...................................................................... -->

<!-- Inventory Module
```

```
         shelf
            item
           sku
           desc
           price

         This module defines a simple inventory item structure
    -->

    <!-- Define the global namespace attributes -->
    <![%Inventory.prefixed;[
    <!ENTITY % Inventory.xmlns.attrib
        "%NS.prefixed.attrib;"
    >
    ]]>
    <!ENTITY % Inventory.xmlns.attrib
        "%NS.prefixed.attrib;
         xmlns  %URI.datatype;  #FIXED '%Inventory.xmlns;'"
    >

    <!ELEMENT %Inventory.shelf.qname;     ( %Inventory.item.qname; )* >
    <!ATTLIST %Inventory.shelf.qname;
         location   CDATA   #IMPLIED
         %Inventory.xmlns.attrib;
    >
    <!ELEMENT %Inventory.item.qname;     ( %Inventory.desc.qname;, %Inventory.sku.qname;, %Inventory.price.qname;) >
    <!ATTLIST %Inventory.item.qname;
         location   CDATA   #IMPLIED
         %Inventory.xmlns.attrib;
    >

    <!ELEMENT %Inventory.desc.qname; ( #PCDATA ) >
    <!ATTLIST %Inventory.desc.qname;
         %Inventory.xmlns.attrib;
    >

    <!ELEMENT %Inventory.sku.qname; ( #PCDATA ) >
    <!ATTLIST %Inventory.sku.qname;
         %Inventory.xmlns.attrib;
    >

    <!ELEMENT %Inventory.price.qname; ( #PCDATA ) >
    <!ATTLIST %Inventory.price.qname;
         %Inventory.xmlns.attrib;
    >

    <!-- end of inventory-1.mod -->
```

D.2.3. Using the module as a stand-alone DTD

It is sometimes desirable to have an XHTML module also usable as a stand alone DTD. A good example of this is our Inventory module above. These items need to be embeddable in an XHTML document, and also need to be available as free-standing documents extracted from a database (for example). The easiest way to accomplish this is to define a DTD file that instantiates the components of your module. Such a DTD would have this structure:

1. Include the XHTML Datatypes module (your qnames module likely uses some of these datatypes).
2. Include the Qnames Module for your module.
3. Define the parameter entity NS.prefixed.attrib to be %MODULE.xmlns.extra.attrib;.
4. Include the Declaration Module(s) for your module.

An example of this for our Inventory module is included below:

```
<!-- ...................................................................... -->
<!-- Inventory Elements DTD ............................................... -->
<!-- file: inventory-1.dtd

     PUBLIC "-//MY COMPANY//DTD XHTML Inventory 1.0//EN"
     SYSTEM "http://www.my.org/DTDs/inventory-1.dtd"

     xmlns:inventory="http://www.my.org/xmlns/inventory"
     ...................................................................... -->

<!-- Inventory Module

     shelf
        item
       sku
       desc
       price

     This module defines a simple inventory item structure
-->

<!-- Bring in the datatypes -->
<!ENTITY % xhtml-datatypes.mod
        SYSTEM "../xhtml-datatypes-1.mod">
<!--         PUBLIC "-//W3C//ENTITIES XHTML Datatypes 1.0//EN"
         "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-datatypes-1.mod" >
         -->
%xhtml-datatypes.mod;

<!-- Bring in the qualified names -->
<!ENTITY % Inventory-qname.mod SYSTEM "inventory-qname-1.mod" >
%Inventory-qname.mod;

<!ENTITY % NS.prefixed.attrib "%Inventory.xmlns.extra.attrib;">

<!ENTITY % Inventory.mod SYSTEM "inventory-1.mod" >
%Inventory.mod;

<!-- end of inventory-1.dtd -->
```

This DTD can then be referenced by documents that use only the elements from your module:

```
<!DOCTYPE shelf SYSTEM "inventory-1.dtd">
<shelf xmlns="http://www.my.org/xmlns/inventory">
    <item>
        <desc>
      this is a description.
    </desc>
        <sku>
      this is the price.
    </sku>
        <price>
      this is the price.
    </price>
    </item>
</shelf>
```

This method permits the definition of elements and attributes that are scoped within their own namespace. It also permits content developers to use the default prefix for the elements and attributes:

```
<!DOCTYPE shelf SYSTEM "inventory-1.dtd" [
    <!ENTITY % Inventory.prefixed "INCLUDE">
]>
<inventory:shelf xmlns:inventory="http://www.my.org/xmlns/inventory">
    <inventory:item>
        <inventory:desc>
      this is a description.
    </inventory:desc>
        <inventory:sku>
      this is the price.
    </inventory:sku>
        <inventory:price>
      this is the price.
    </inventory:price>
    </inventory:item>
</inventory:shelf>
```

Finally, a document instance can use a different XML Namespace prefix by redeclaring it in the DOCTYPE header and its internal subset:

```
<!DOCTYPE shelf SYSTEM "inventory-1.dtd" [
    <!ENTITY % Inventory.prefixed "INCLUDE">
    <!ENTITY % Inventory.prefix "i">
]>
<i:shelf xmlns:i="http://www.my.org/xmlns/inventory">
    <i:item>
        <i:desc>
      this is a description.
    </i:desc>
        <i:sku>
      this is the price.
    </i:sku>
        <i:price>
      this is the price.
    </i:price>
    </i:item>
</i:shelf>
```

D.2.4. Namespace Idiosyncracies

While the approach defined here permits the definition of markup languages that are XML and XML Namespaces conforming, some behaviors defined by the XML Namespaces specification are not supported:

1. XML Namespaces permits the redeclaration of the xmlns attribute for a namespace at any point in the tree. It further permits this redeclaration to switch between namespace defaulting and prefixed usage, and permits the changing of the prefix. The method defined in this document does not permit this. Throughout a document instance a given namespace must continue to use the same namespace prefix (when prefixing is used), or must continue to be used in the default scope.

2. When using XML Namespace defaulting, it is legal to rely upon the DTD of the document to inform parsers of the namespace of elements. However, since namespace aware processors are not required to include the DTD when evaluating a document, content developers should declare the XML Namespace of an element whenever the namespace changes:

   ```
   ...
   <p>
      <myelement xmlns="..." />
   </p>
   ```

E. Developing DTDs with defined and extended modules

This section is **informative**.

The primary purpose of defining XHTML modules and a general modularization methodology is to ease the development of document types that are based upon XHTML. These document types may extend XHTML by integrating additional capabilities (e.g., [SMIL] [p.168]), or they may define a subset of XHTML for use in a specialized device. This section describes the techniques that document type designers must use in order to take advantage of the XML DTD implementation of this modularization architecture. It does this by applying the XHTML Modularization techniques in progressively more complex ways, culminating in the creation of a complete document type from disparate modules.

Note that in no case do these examples require the modification of the XHTML-provided module *files* themselves. The XHTML module files are completely parameterized, so that it is possible through separate module definitions and *driver files* to customize the definition and the content model of each element and each element’s hierarchy.

Finally, remember that most users of XHTML are *not* expected to be DTD authors. DTD authors are generally people who are defining specialized markup that will improve the readability, simplify the rendering of a document, or ease machine-processing of documents, or they are client designers that need to define the specialized DTD for their specific client. Consider these cases:

- An organization is providing subscriber’s information via a web interface. The organization stores its subscriber information in an XML-based database. One way to report that information out from the database to the web is to embed the XML records from the database directly in the XHTML document. While it is possible to merely embed the records, the organization could define a DTD module that describes the records, attach that module to an XHTML DTD, and thereby create a complete DTD for the pages. The organization can then access the data within the new elements via the Document Object Model [DOM] [p.167] , validate the documents, provide style definitions for the elements that cascade using Cascading Style Sheets [CSS2] [p.167] , etc. By taking the time to define the structure of their data and create a DTD using the processes defined in this section, the organization can realize the full benefits of XML.

- An Internet client developer is designing a specialized device. That device will only support a subset of XHTML, and the devices will always access the Internet via a proxy server that validates content before passing it on to the client (to minimize error handling on the client). In order to ensure that the content is valid, the developer creates a DTD that is a subset of XHTML using the processes defined in this section. They then use the new DTD in their proxy server and in their devices, and also make the DTD available to content developers so that developers can validate their content before making it available. By performing a few simple steps, the client developer can use the architecture defined in this document to greatly ease their DTD development cost *and* ensure that they are fully supporting the subset of XHTML that they choose to include.

E.1. Defining additional attributes

In some cases, an extension to XHTML can be as simple as additional attributes. Attributes can be added to an element just by specifying an additional ATTLIST for the element, for example:

```
<!ATTLIST %a.qname;
      %MyModule.pfx;myattr   CDATA        #IMPLIED
>
```

would add the "myattr" attribute, with an optional prefix defined by "%MyModule.pfx", with a value type of CDATA, to the "a" element. This works because XML permits the definition or extension of the attribute list for an element at any point in a DTD. *For a discussion of qualified names and namespace prefixes, see Defining the Namespace of a Module [p.56] .*

Naturally, adding an attribute to a DTD does not mean that any new behavior is defined for arbitrary clients. However, a content developer could use an extra attribute to store information that is accessed by associated scripts via the Document Object Model (for example).

E.2. Defining additional elements

Defining additional elements is only slightly more complicated than defining additional attributes. Basically, DTD authors should write the element declaration for each element:

```
<!ENTITY % MyModule.myelement.qname  "%MyModule.pfx;myelement" >
<!ENTITY % MyModule.myotherelement.qname  "%MyModule.pfx;myotherelement" >
<!ELEMENT %MyModule.myelement.qname;
           ( #PCDATA | %MyModule.myotherelement.qname; )* >
<!ATTLIST %MyModule.myelement.qname;
          myattribute    CDATA    #IMPLIED
>

<!ELEMENT %MyModule.myotherelement.qname; EMPTY >
```

After the elements are defined, they need to be integrated into the content model. Strategies for integrating new elements or sets of elements into the content model are addressed in the next section.

E.3. Defining the content model for a collection of modules

Since the content model of XHTML modules is fully parameterized, DTD authors may modify the content model for every element in every module. The details of the DTD module interface are defined in Building DTD Modules [p.55] . Basically there are two ways to approach this modification:

1. Re-define the ".content" parameter entity for each element.
2. Re-define one or more of the global content model entities (normally the ".extras" parameter entity).

The strategy taken will depend upon the nature of the modules being combined and the nature of the elements being integrated. The remainder of this section describes techniques for integrating two different classes of modules.

E.3.1. Integrating a stand-alone module into XHTML

When a module (and remember, a module can be a collection of other modules) contains elements that only reference each other in their content model, it is said to be "internally complete". As such, the module can be used on its own; (for example, you could define a DTD that was just that module, and use one of its elements as the root element). Integrating such a module into XHTML is a three step process:

1. Decide what element(s) can be thought of as the root(s) of the new module.
2. Decide where these elements need to attach in the XHTML content tree.
3. Then, for each attachment point in the content tree, add the root element(s) to the content definition for the XHTML elements.

Consider attaching the elements defined above [p.64] . In that example, the element `myelement` is the root. To attach this element under the `img` element, and only the `img` element, of XHTML, the following would work:

```
<!ENTITY % img.content "( %MyModule.myelement.qname; )*">
```

A DTD defined with this content model would allow a document like the following fragment:

```
<img src="...">
<myml:myelement >This is content of a locally defined element</myml:myelement>
</img>
```

It is important to note that normally the `img` element has a content model of `EMPTY`. By adding myelement to that content model, we are really just replacing `EMPTY` with `myelement`. In the case of other elements that already have content models defined, the addition of an element would require the restating of the existing content model in addition to `myelement`.

E.3.2. Mixing a new module throughout the modules in XHTML

Extending the example above, to attach this module everywhere that the `%Flow.mix` content model group is permitted, would require something like the following:

```
<!ENTITY % Misc.extra
     "| %script.qname; | %noscript.qname; | %MyModule.myelement.qname;" >
```

Since the %Misc.extra content model class is used in the %Misc.class parameter entity, and that parameter entity is used throughout the XHTML Modules, the new module would become available throughout an extended XHTML document type.

E.4. Creating a new DTD

So far the examples in this section have described the methods of extending XHTML and XHTML’s content model. Once this is done, the next step is to collect the modules that comprise the DTD into a single DTD driver, incorporating the new definitions so that they override and augment the basic XHTML definitions as appropriate.

E.4.1. Creating a simple DTD

Using the trivial example above, it is possible to define a new DTD that uses and extends the XHTML modules pretty easily. First, define the new elements and their content model in a module:

```
<!-- File: smallml-model-1.mod -->

<!-- Declare a Parameter Entity (PE) that defines any external namespaces
     that are used by this module -->

<!-- Set the PE that is used in every ATTLIST in this module
     NS.prefixed.attrib is initialized in the xhtml-qname module, and
     SimpleML.ns.noprefix.attrib is initialized in the SimpleML DTD driver
     file.-->
<!ENTITY % SimpleML.ns.attrib
  "%NS.prefixed.attrib;
   %SimpleML.ns.noprefix.attrib;"
>

<!ENTITY % SimpleML.Element.qname "%SimpleML.pfx;element" >
<!ENTITY % SimpleML.Otherelement.qname "%SimpleML.pfx;otherelement" >

<!ELEMENT %SimpleML.Element.qname;          ( #PCDATA | %SimpleML.Otherelement.qname; )* >
<!ATTLIST %SimpleML.Element.qname;
          myattribute   CDATA  #IMPLIED
          %SimpleML.ns.attrib;
>
<!ELEMENT %SimpleML.Otherelement.qname; EMPTY >
<!ATTLIST %SimpleML.Otherelement.qname;
          %SimpleML.ns.attrib;
>

<!ENTITY % SimpleML.Img.myattr.qname "%SimpleML.pfx;myattr" >
<!ATTLIST %img.qname;
          %SimpleML.Img.myattr.qname;  CDATA  #IMPLIED
>

<!-- Add our elements to the XHTML content model -->
<!ENTITY % Misc.extra
     "| %script.qname; | %noscript.qname; | %SimpleML.Element.qname;" >

<!-- Now bring in the XHTML 1.1 content model -->
```

```
<!ENTITY % xhtml11-model.mod
     PUBLIC "-//W3C//ENTITIES XHTML 1.1 Document Model 1.0//EN"
            "http://www.w3.org/TR/xhtml11/DTD/xhtml11-model.mod" >
%xhtml11-model.mod;
```

Next, define the DTD driver for the new language:

```
<!-- file: simpleml-1_0.dtd -->

<!-- Bring in the XHTML datatypes -->
<!ENTITY % xhtml-datatypes.mod
     PUBLIC "-//W3C//ENTITIES XHTML Datatypes 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-datatypes-1.mod" >
%xhtml-datatypes.mod;

<!-- By default, disable prefixing of new module -->
<!ENTITY % SimpleML.prefixed "IGNORE" >

<!-- If this module’s namespace is prefixed -->
<![%SimpleML.prefixed;[
  <!ENTITY % SimpleML.prefix "simpleml" >
  <!ENTITY % SimpleML.pfx  "%SimpleML.prefix;:" >
  <!ENTITY % SimpleML.prefixed.extras.attrib
       "xmlns:%SimpleML.prefix; %URI.datatype; #FIXED ’SimpleMLns’" >
  <!ENTITY % SimpleML.ns.noprefix.attrib ’’ >
]]>

<!-- If this module’s namespace is not prefixed, set the Parameter Entities
     (PEs) as needed -->
<!ENTITY % SimpleML.pfx  "" >
<!ENTITY % SimpleML.ns.prefixed.attrib "" >
<!ENTITY % SimpleML.ns.noprefix.attrib
     "xmlns  %URI.datatype; #FIXED ’SimpleMLns’" >

<!-- Set up the NS.prefixed.extras.attrib -->
<!ENTITY % NS.prefixed.extras.attrib
     "%SimpleML.prefixed.extras.attrib;" >

<!-- Set the content model for our language -->
<!ENTITY % xhtml-model.mod
     SYSTEM "simpleml-model-1.mod" >
<!-- Instantiate xhtml11’s DTD to do all the work -->
<!ENTITY % xhtml11.dtd
     PUBLIC "-//W3C//DTD XHTML 1.1//EN"
            "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd" >
%xhtml11.dtd;
```

When using this DTD, it is possible to enable the use of XML Namespace prefixes. When so doing, the start of a document using this new DTD might look like:

```
<!DOCTYPE html SYSTEM "simpleml-1_0.dtd" [
  <!ENTITY % SimpleML.prefixed "INCLUDE">
]>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:simpleml="SimpleMLns" >
<head>
```

```
<title>An example using defaults</title>
</head>
<body>
<p>This is content in the XHTML namespace</p>
<simpleml:element>
  This is content in the SimpleML namespace.
  <simpleml:otherelement />
</simpleml:element>
<p><img src="missing" alt="Missing image" simpleml:myattr="value"/></p>
</body>
</html>
```

E.4.2. Creating a DTD by extending XHTML

Next, there is the situation where a complete, additional, and complex module is added to XHTML (or to a subset of XHTML). In essence, this is the same as in the trivial example above, the only difference being that the module being added is incorporated in the DTD by reference rather than explicitly including the new definitions in the DTD.

One such complex module is the DTD for [MathML] [p.167] . In order to combine MathML and XHTML into a single DTD, an author would just decide where MathML content should be legal in the document, and add the MathML root element to the content model at that point. First, define a content model module that instantiates the MathML DTD and connects it to the content model:

```
<!-- File: mathml-model.mod -->
<!ENTITY % XHTML1-math
     PUBLIC "-//W3C//MathML 2.0//EN"
            "http://www.w3.org/TR/2000/WD-MathML2-20000328/dtd/mathml2.dtd >
%XHTML1-math;

<!ENTITY % Inlspecial.extra
     "%a.qname; | %img.qname; | %object.qname; | %map.qname;
      | %Mathml.Math.qname;" >
```

Next, define a DTD driver that identifies our new content model module as the content model for the DTD, and hands off processing to the XHTML 1.1 driver (for example):

```
<!-- File: xhtml-mathml.dtd -->
<!ENTITY % xhtml-model.mod
      SYSTEM "mathml-model.mod" >
<!ENTITY % xhtml11.dtd
     PUBLIC "-//W3C//XHTML 1.1//EN"
            "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd" >
%xhtml11.dtd;
```

E.4.3. Creating a DTD by removing and replacing XHTML modules

Another way in which DTD authors may use XHTML modules is to define a DTD that is a subset of an XHTML family document type (because, for example, they are building devices or software that only supports a subset of XHTML). Doing this is only slightly more complex than the previous example. The basic steps to follow are:

1. Take an XHTML family DTD as the basis of the new document type (we will use XHTML 1.1).
2. Select the modules to remove from that DTD.
3. Define a new DTD that "IGNOREs" the modules.

For example, consider a device that uses XHTML modules, but without forms or tables. The DTD for such a device would look like this:

```
<!-- File: xhtml-simple.dtd -->
<!ENTITY % xhtml-form.module "IGNORE" >
<!ENTITY % xhtml-table.module "IGNORE" >

<!ENTITY % xhtml11.mod
     PUBLIC "-//W3C//DTD XHTML 1.1//EN"
            "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd" >
%xhtml11.mod;
```

Note that this does not actually modify the content model for the XHTML 1.1 DTD. However, since XML ignores elements in content models that are not defined, the form and table elements are dropped from the model automatically.

E.4.4. Creating a new DTD

Finally, some DTD authors may wish to start from scratch, using the XHTML Modularization framework as a toolkit for building a new markup language. This language must be made up of the minimal, required modules from XHTML. It may also contain other XHTML-defined modules or any other module that the author wishes to employ. In this example, we will take the XHTML required modules, add some XHTML-defined modules, and also add in the module we defined above.

The first step is to use the XHTML-provided template for a new qualified names module, modified to define the qualified names and namespace for our new elements.

```
<!-- file: myml-qname-1.mod -->

<!-- Bring in the datatypes -->
<!ENTITY % MyML-datatypes.mod
         PUBLIC "-//W3C//ENTITIES XHTML Datatypes 1.0//EN"
         "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-datatypes-1.mod" >
%MyML-datatypes.mod;

<!-- Declare the actual namespace of this module -->
<!ENTITY % MyML.xmlns "http://www.my.org/xmlns/myml" >

<!-- Declare the default prefix for this module -->
<!ENTITY % MyML.prefix "myml" >

<!-- By default, disable prefixing of this module -->
<!ENTITY % NS.prefixed "IGNORE" >
<!ENTITY % MyML.prefixed "%NS.prefixed;" >

<!-- If this module’s namespace is prefixed -->
```

```
<![%MyML.prefixed;[
  <!ENTITY % MyML.pfx  "%MyML.prefix;:" >
]]>
<!ENTITY % MyML.pfx  "" >

<!-- Declare a Parameter Entity (PE) that defines any external namespaces
     that are used by this module -->
<!ENTITY % MyML.ns.extra.attrib "" >

<!-- Declare a PE that defines the xmlns attributes for use by by MyML. -->
<![%MyML.prefixed;[
<!ENTITY % MyML.xmlns.attrib
   "xmlns:%MyML.prefix;  %URI.datatype;  #FIXED '%MyML.xmlns;'
   %MyML.xmlns.extra.attrib;"
>
]]>
<!ENTITY % MyML.xmlns.attrib
   "xmlns   %URI.datatype;  #FIXED '%MyML.xmlns;'
    %MyML.xmlns.extra.attrib;"
>

<!-- Make sure that the MyML namespace attributes are included on the XHTML
     attribute set -->
<![%NS.prefixed;[
<!ENTITY % XHTML.xmlns.extra.attrib
    "%MyML.xmlns.attrib;" >
]]>
<!ENTITY % XHTML.xmlns.extra.attrib
    "%XLINK.xmlns.attrib;"
>
<!-- Now declare the element names -->

<!ENTITY % MyML.myelement.qname "%MyML.pfx;myelement" >
<!ENTITY % MyML.myotherelement.qname "%MyML.pfx;myotherelement" >
```

Next, define a module that defines the elements and attributes using the XHTML provided template.

```
<!-- ...................................................................... -->
<!-- My Elements Module ................................................... -->
<!-- file: myml-elements-1_0.mod

     PUBLIC "-//MY COMPANY//ELEMENTS XHTML MyML Elements 1.0//EN"
     SYSTEM "http://www.my.org/DTDs/myml-elements-1_0.mod"

     xmlns:myml="http://www.my.org/DTDs/myml-1_0.dtd"
     ...................................................................... -->

<!-- My Elements Module

     myelement
     myotherelement

     This module has no purpose other than to provide structure for some
     PCDATA content.
-->
```

```
<!-- First define the global namespace attributes -->
<!ENTITY % MyML.ns.attrib
     "%NS.prefixed.attrib;
      %MyML.ns.noprefix.attrib;"
>

<!ELEMENT %MyML.Myelement.qname;     ( #PCDATA | %MyML.Myotherelement.qname; )* >
<!ATTLIST %MyML.Myelement.qname;
     myattribute    CDATA   #IMPLIED
     %MyML.ns.attrib;
>

<!ELEMENT %MyML.Myotherelement.qname; EMPTY >
<!ATTLIST %MyML.Myotherelement.qname;
      %MyML.ns.attrib;
>

<!ENTITY % MyML.Img.myattr.qname "%MyML.pfx;myattr" >
<!ATTLIST %img.qname;
      %MyML.Img.myattr.qname;   CDATA  #IMPLIED
>

<!-- end of myml-elements-1_0.mod -->
```

Finally, use the XHTML-provided template for a new DTD, modified as appropriate for our new markup language:

```
<!-- ....................................................................... -->
<!-- MYML DTD  ............................................................. -->
<!-- file: myml-1_0.dtd -->

<!-- This is the DTD driver for myml 1.0.

     Please use this formal public identifier to identify it:

         "-//MY COMPANY//DTD XHTML MYML 1.0//EN"

     And this namespace for myml-unique elements:

         xmlns:myml="http://www.my.org/xmlns/myml"
-->
<!ENTITY % XHTML.version  "-//MY COMPANY//DTD XHTML MYML 1.0//EN" >

<!-- Bring in the qualified names for the new module -->
<!ENTITY % MyML-qname.mod
     SYSTEM "myml-qname-1.mod" >
%MyML-qname.mod;

<!-- There are no extra namespaces to include -->
<!ENTITY % NS.prefixed.extras.attrib "" >

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

<!-- Define the Content Model for the framework to use -->
```

```
<!ENTITY % xhtml-model.mod
     SYSTEM "myml-model-1.mod" >

<!-- Disable bidirectional text support -->
<!ENTITY % XHTML.bidi  "INCLUDE" >

<!-- Bring in the XHTML Framework -->
<!ENTITY % xhtml-framework.mod
     PUBLIC "-//W3C//ENTITIES XHTML Modular Framework 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-framework-1.mod" >
%xhtml-framework.mod;

<!-- Basic Text Module (Required)  ............................... -->
<!ENTITY % xhtml-text.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Basic Text 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-text-1.mod" >
%xhtml-text.mod;

<!-- Hypertext Module (required) ................................. -->
<!ENTITY % xhtml-hypertext.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Hypertext 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-hypertext-1.mod" >
%xhtml-hypertext.mod;

<!-- Lists Module (required)  .................................... -->
<!ENTITY % xhtml-list.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Lists 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-list-1.mod" >
%xhtml-list.mod;

<!-- My Elements Module   ........................................ -->
<!ENTITY % MyML-elements.mod
     SYSTEM "myml-elements-1.mod" >
%MyML-elements.mod;

<!-- XHTML Images module  ........................................ -->
<!ENTITY % xhtml-image.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Images 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-image-1.mod" >
%xhtml-image.mod;

<!-- Document Metainformation Module  ............................ -->
<!ENTITY % xhtml-meta.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Metainformation 1.0//EN"
            "xhtml-meta-1.mod" >
%xhtml-meta.mod;

<!-- Document Structure Module (required)  ....................... -->
<!ENTITY % xhtml-struct.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Document Structure 1.0//EN"
            "http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-struct-1.mod" >
%xhtml-struct.mod;
```

E.5. Using the new DTD

Once a new DTD has been developed, it can be used in any document. Using the DTD is as simple as just referencing it in the DOCTYPE declaration of a document:

```
Module DTD/examples/myml-default.html not found!
```

The document can also use the elements outside of the XHTML namespace by prefixing them:

```
<!DOCTYPE html SYSTEM "myml-1_0.dtd" [
  <!ENTITY % MyML.prefixed "INCLUDE" >
]>
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
<title>An example using defaults</title>
</head>
<body>
<p>This is content in the XHTML namespace</p>
<myml:myelement>
  This is content in the SimpleML namespace.
  <myml:myotherelement />
</myml:myelement>
<p><img src="missing" alt="Missing image" myml:myattr="value"/></p>
</body>
</html>
```

F. XHTML DTD Module Implementations

This appendix is *normative*.

This appendix will contain implementations of the modules defined in XHTML Abstract Modules [p.27] via XML DTDs. These module implementations can be used by XHTML Family Document Types.

F.1. XHTML Character Entities

XHTML DTDs make available a standard collection of named character entities. Those entities are defined in this section.

F.1.1. XHTML Latin 1 Character Entities

```
<!-- ...................................................................... -->
<!-- XML-compatible ISO Latin 1 Character Entity Set for XHTML ............ -->
<!-- file: xhtml-lat1.ent

     Typical invocation:

       <!ENTITY % xhtml-lat1
           PUBLIC "-//W3C//ENTITIES Latin 1 for XHTML//EN"
                  "xhtml-lat1.ent" >
       %xhtml-lat1;

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

     PUBLIC "-//W3C//ENTITIES Latin 1 for XHTML//EN"
     SYSTEM "xhtml-lat1.ent"

     Revision:  $Id: xhtml-lat1.ent,v 1.2 2000/02/03 15:59:28 ahby Exp $ SMI

     Portions (C) International Organization for Standardization 1986:
     Permission to copy in any form is granted for use with conforming
     SGML systems and applications as defined in ISO 8879, provided
     this notice is included in all copies.
-->

<!ENTITY nbsp   "&#160;" ><!-- no-break space = non-breaking space,
                                  U+00A0 ISOnum -->
<!ENTITY iexcl  "&#161;" ><!-- inverted exclamation mark,
                                  U+00A1 ISOnum -->
<!ENTITY cent   "&#162;" ><!-- cent sign,
                                  U+00A2 ISOnum -->
<!ENTITY pound  "&#163;" ><!-- pound sign,
                                  U+00A3 ISOnum -->
<!ENTITY curren "&#164;" ><!-- currency sign,
                                  U+00A4 ISOnum -->
<!ENTITY yen    "&#165;" ><!-- yen sign = yuan sign,
                                  U+00A5 ISOnum -->
<!ENTITY brvbar "&#166;" ><!-- broken bar = broken vertical bar,
                                  U+00A6 ISOnum -->
```

```
<!ENTITY sect   "&#167;" ><!-- section sign,
                                  U+00A7 ISOnum -->
<!ENTITY uml    "&#168;" ><!-- diaeresis = spacing diaeresis,
                                  U+00A8 ISOdia -->
<!ENTITY copy   "&#169;" ><!-- copyright sign,
                                  U+00A9 ISOnum -->
<!ENTITY ordf   "&#170;" ><!-- feminine ordinal indicator,
                                  U+00AA ISOnum -->
<!ENTITY laquo  "&#171;" ><!-- left-pointing double angle quotation mark
                                  = left pointing guillemet,
                                  U+00AB ISOnum -->
<!ENTITY not    "&#172;" ><!-- not sign,
                                  U+00AC ISOnum -->
<!ENTITY shy    "&#173;" ><!-- soft hyphen = discretionary hyphen,
                                  U+00AD ISOnum -->
<!ENTITY reg    "&#174;" ><!-- registered sign = registered trade mark sign,
                                  U+00AE ISOnum -->
<!ENTITY macr   "&#175;" ><!-- macron = spacing macron = overline
                                  = APL overbar,
                                  U+00AF ISOdia -->
<!ENTITY deg    "&#176;" ><!-- degree sign,
                                  U+00B0 ISOnum -->
<!ENTITY plusmn "&#177;" ><!-- plus-minus sign = plus-or-minus sign,
                                  U+00B1 ISOnum -->
<!ENTITY sup2   "&#178;" ><!-- superscript two = superscript digit two
                                  = squared,
                                  U+00B2 ISOnum -->
<!ENTITY sup3   "&#179;" ><!-- superscript three = superscript digit three
                                  = cubed,
                                  U+00B3 ISOnum -->
<!ENTITY acute  "&#180;" ><!-- acute accent = spacing acute,
                                  U+00B4 ISOdia -->
<!ENTITY micro  "&#181;" ><!-- micro sign,
                                  U+00B5 ISOnum -->
<!ENTITY para   "&#182;" ><!-- pilcrow sign = paragraph sign,
                                  U+00B6 ISOnum -->
<!ENTITY middot "&#183;" ><!-- middle dot = Georgian comma
                                  = Greek middle dot,
                                  U+00B7 ISOnum -->
<!ENTITY cedil  "&#184;" ><!-- cedilla = spacing cedilla,
                                  U+00B8 ISOdia -->
<!ENTITY sup1   "&#185;" ><!-- superscript one = superscript digit one,
                                  U+00B9 ISOnum -->
<!ENTITY ordm   "&#186;" ><!-- masculine ordinal indicator,
                                  U+00BA ISOnum -->
<!ENTITY raquo  "&#187;" ><!-- right-pointing double angle quotation mark
                                  = right pointing guillemet,
                                  U+00BB ISOnum -->
<!ENTITY frac14 "&#188;" ><!-- vulgar fraction one quarter
                                  = fraction one quarter,
                                  U+00BC ISOnum -->
<!ENTITY frac12 "&#189;" ><!-- vulgar fraction one half
                                  = fraction one half,
                                  U+00BD ISOnum -->
<!ENTITY frac34 "&#190;" ><!-- vulgar fraction three quarters
                                  = fraction three quarters,
                                  U+00BE ISOnum -->
```

```
<!ENTITY iquest "&#191;" ><!-- inverted question mark
                                  = turned question mark,
                                  U+00BF ISOnum -->
<!ENTITY Agrave "&#192;" ><!-- latin capital letter A with grave
                                  = latin capital letter A grave,
                                  U+00C0 ISOlat1 -->
<!ENTITY Aacute "&#193;" ><!-- latin capital letter A with acute,
                                  U+00C1 ISOlat1 -->
<!ENTITY Acirc  "&#194;" ><!-- latin capital letter A with circumflex,
                                  U+00C2 ISOlat1 -->
<!ENTITY Atilde "&#195;" ><!-- latin capital letter A with tilde,
                                  U+00C3 ISOlat1 -->
<!ENTITY Auml   "&#196;" ><!-- latin capital letter A with diaeresis,
                                  U+00C4 ISOlat1 -->
<!ENTITY Aring  "&#197;" ><!-- latin capital letter A with ring above
                                  = latin capital letter A ring,
                                  U+00C5 ISOlat1 -->
<!ENTITY AElig  "&#198;" ><!-- latin capital letter AE
                                  = latin capital ligature AE,
                                  U+00C6 ISOlat1 -->
<!ENTITY Ccedil "&#199;" ><!-- latin capital letter C with cedilla,
                                  U+00C7 ISOlat1 -->
<!ENTITY Egrave "&#200;" ><!-- latin capital letter E with grave,
                                  U+00C8 ISOlat1 -->
<!ENTITY Eacute "&#201;" ><!-- latin capital letter E with acute,
                                  U+00C9 ISOlat1 -->
<!ENTITY Ecirc  "&#202;" ><!-- latin capital letter E with circumflex,
                                  U+00CA ISOlat1 -->
<!ENTITY Euml   "&#203;" ><!-- latin capital letter E with diaeresis,
                                  U+00CB ISOlat1 -->
<!ENTITY Igrave "&#204;" ><!-- latin capital letter I with grave,
                                  U+00CC ISOlat1 -->
<!ENTITY Iacute "&#205;" ><!-- latin capital letter I with acute,
                                  U+00CD ISOlat1 -->
<!ENTITY Icirc  "&#206;" ><!-- latin capital letter I with circumflex,
                                  U+00CE ISOlat1 -->
<!ENTITY Iuml   "&#207;" ><!-- latin capital letter I with diaeresis,
                                  U+00CF ISOlat1 -->
<!ENTITY ETH    "&#208;" ><!-- latin capital letter ETH,
                                  U+00D0 ISOlat1 -->
<!ENTITY Ntilde "&#209;" ><!-- latin capital letter N with tilde,
                                  U+00D1 ISOlat1 -->
<!ENTITY Ograve "&#210;" ><!-- latin capital letter O with grave,
                                  U+00D2 ISOlat1 -->
<!ENTITY Oacute "&#211;" ><!-- latin capital letter O with acute,
                                  U+00D3 ISOlat1 -->
<!ENTITY Ocirc  "&#212;" ><!-- latin capital letter O with circumflex,
                                  U+00D4 ISOlat1 -->
<!ENTITY Otilde "&#213;" ><!-- latin capital letter O with tilde,
                                  U+00D5 ISOlat1 -->
<!ENTITY Ouml   "&#214;" ><!-- latin capital letter O with diaeresis,
                                  U+00D6 ISOlat1 -->
<!ENTITY times  "&#215;" ><!-- multiplication sign,
                                  U+00D7 ISOnum -->
<!ENTITY Oslash "&#216;" ><!-- latin capital letter O with stroke
                                  = latin capital letter O slash,
                                  U+00D8 ISOlat1 -->
```

```
<!ENTITY Ugrave "&#217;" ><!-- latin capital letter U with grave,
                                     U+00D9 ISOlat1 -->
<!ENTITY Uacute "&#218;" ><!-- latin capital letter U with acute,
                                     U+00DA ISOlat1 -->
<!ENTITY Ucirc  "&#219;" ><!-- latin capital letter U with circumflex,
                                     U+00DB ISOlat1 -->
<!ENTITY Uuml   "&#220;" ><!-- latin capital letter U with diaeresis,
                                     U+00DC ISOlat1 -->
<!ENTITY Yacute "&#221;" ><!-- latin capital letter Y with acute,
                                     U+00DD ISOlat1 -->
<!ENTITY THORN  "&#222;" ><!-- latin capital letter THORN,
                                     U+00DE ISOlat1 -->
<!ENTITY szlig  "&#223;" ><!-- latin small letter sharp s = ess-zed,
                                     U+00DF ISOlat1 -->
<!ENTITY agrave "&#224;" ><!-- latin small letter a with grave
                                     = latin small letter a grave,
                                     U+00E0 ISOlat1 -->
<!ENTITY aacute "&#225;" ><!-- latin small letter a with acute,
                                     U+00E1 ISOlat1 -->
<!ENTITY acirc  "&#226;" ><!-- latin small letter a with circumflex,
                                     U+00E2 ISOlat1 -->
<!ENTITY atilde "&#227;" ><!-- latin small letter a with tilde,
                                     U+00E3 ISOlat1 -->
<!ENTITY auml   "&#228;" ><!-- latin small letter a with diaeresis,
                                     U+00E4 ISOlat1 -->
<!ENTITY aring  "&#229;" ><!-- latin small letter a with ring above
                                     = latin small letter a ring,
                                     U+00E5 ISOlat1 -->
<!ENTITY aelig  "&#230;" ><!-- latin small letter ae
                                     = latin small ligature ae,
                                     U+00E6 ISOlat1 -->
<!ENTITY ccedil "&#231;" ><!-- latin small letter c with cedilla,
                                     U+00E7 ISOlat1 -->
<!ENTITY egrave "&#232;" ><!-- latin small letter e with grave,
                                     U+00E8 ISOlat1 -->
<!ENTITY eacute "&#233;" ><!-- latin small letter e with acute,
                                     U+00E9 ISOlat1 -->
<!ENTITY ecirc  "&#234;" ><!-- latin small letter e with circumflex,
                                     U+00EA ISOlat1 -->
<!ENTITY euml   "&#235;" ><!-- latin small letter e with diaeresis,
                                     U+00EB ISOlat1 -->
<!ENTITY igrave "&#236;" ><!-- latin small letter i with grave,
                                     U+00EC ISOlat1 -->
<!ENTITY iacute "&#237;" ><!-- latin small letter i with acute,
                                     U+00ED ISOlat1 -->
<!ENTITY icirc  "&#238;" ><!-- latin small letter i with circumflex,
                                     U+00EE ISOlat1 -->
<!ENTITY iuml   "&#239;" ><!-- latin small letter i with diaeresis,
                                     U+00EF ISOlat1 -->
<!ENTITY eth    "&#240;" ><!-- latin small letter eth,
                                     U+00F0 ISOlat1 -->
<!ENTITY ntilde "&#241;" ><!-- latin small letter n with tilde,
                                     U+00F1 ISOlat1 -->
<!ENTITY ograve "&#242;" ><!-- latin small letter o with grave,
                                     U+00F2 ISOlat1 -->
<!ENTITY oacute "&#243;" ><!-- latin small letter o with acute,
                                     U+00F3 ISOlat1 -->
```

```
<!ENTITY ocirc  "&#244;" ><!-- latin small letter o with circumflex,
                                  U+00F4 ISOlat1 -->
<!ENTITY otilde "&#245;" ><!-- latin small letter o with tilde,
                                  U+00F5 ISOlat1 -->
<!ENTITY ouml   "&#246;" ><!-- latin small letter o with diaeresis,
                                  U+00F6 ISOlat1 -->
<!ENTITY divide "&#247;" ><!-- division sign,
                                  U+00F7 ISOnum -->
<!ENTITY oslash "&#248;" ><!-- latin small letter o with stroke,
                                  = latin small letter o slash,
                                  U+00F8 ISOlat1 -->
<!ENTITY ugrave "&#249;" ><!-- latin small letter u with grave,
                                  U+00F9 ISOlat1 -->
<!ENTITY uacute "&#250;" ><!-- latin small letter u with acute,
                                  U+00FA ISOlat1 -->
<!ENTITY ucirc  "&#251;" ><!-- latin small letter u with circumflex,
                                  U+00FB ISOlat1 -->
<!ENTITY uuml   "&#252;" ><!-- latin small letter u with diaeresis,
                                  U+00FC ISOlat1 -->
<!ENTITY yacute "&#253;" ><!-- latin small letter y with acute,
                                  U+00FD ISOlat1 -->
<!ENTITY thorn  "&#254;" ><!-- latin small letter thorn with,
                                  U+00FE ISOlat1 -->
<!ENTITY yuml   "&#255;" ><!-- latin small letter y with diaeresis,
                                  U+00FF ISOlat1 -->
<!-- end of xhtml-lat1.ent -->
```

F.1.2. XHTML Special Characters

```
<!-- ...................................................................... -->
<!-- XML-compatible ISO Special Character Entity Set for XHTML ............ -->
<!-- file: xhtml-lat1.ent

     Typical invocation:

       <!ENTITY % xhtml-special
           PUBLIC "-//W3C//ENTITIES Special for XHTML//EN"
                  "xhtml-special.ent" >
       %xhtml-special;

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

     PUBLIC "-//W3C//ENTITIES Special for XHTML//EN"
     SYSTEM "xhtml-special.ent"

     Revision:  $Id: xhtml-special.ent,v 1.2 2000/02/03 15:59:28 ahby Exp $ SMI

     Portions (C) International Organization for Standardization 1986:
     Permission to copy in any form is granted for use with conforming
     SGML systems and applications as defined in ISO 8879, provided
     this notice is included in all copies.
-->

<!-- Relevant ISO entity set is given unless names are newly introduced.
     New names (i.e., not in ISO 8879 list) do not clash with any
     existing ISO 8879 entity names. ISO 10646 character numbers
     are given for each character, in hex. CDATA values are decimal
     conversions of the ISO 10646 values and refer to the document
     character set. Names are Unicode 2.0 names.
-->

<!-- C0 Controls and Basic Latin -->
<!ENTITY quot    "&#34;" ><!-- quotation mark = APL quote, U+0022 ISOnum -->
<!ENTITY amp     "&#38;" ><!-- ampersand, U+0026 ISOnum -->
<!ENTITY lt      "&#60;" ><!-- less-than sign, U+003C ISOnum -->
<!ENTITY gt      "&#62;" ><!-- greater-than sign, U+003E ISOnum -->

<!-- Latin Extended-A -->
<!ENTITY OElig   "&#338;" ><!-- latin capital ligature OE, U+0152 ISOlat2 -->
<!ENTITY oelig   "&#339;" ><!-- latin small ligature oe, U+0153 ISOlat2 -->

<!-- ligature is a misnomer, this is a separate character in some languages -->
<!ENTITY Scaron  "&#352;" ><!-- latin capital letter S with caron,
                                      U+0160 ISOlat2 -->
<!ENTITY scaron  "&#353;" ><!-- latin small letter s with caron,
                                      U+0161 ISOlat2 -->
<!ENTITY Yuml    "&#376;" ><!-- latin capital letter Y with diaeresis,
                                      U+0178 ISOlat2 -->

<!-- Spacing Modifier Letters -->
<!ENTITY circ    "&#710;" ><!-- modifier letter circumflex accent,
                                      U+02C6 ISOpub -->
<!ENTITY tilde   "&#732;" ><!-- small tilde, U+02DC ISOdia -->

<!-- General Punctuation -->
<!ENTITY ensp    "&#8194;" ><!-- en space, U+2002 ISOpub -->
<!ENTITY emsp    "&#8195;" ><!-- em space, U+2003 ISOpub -->
<!ENTITY thinsp  "&#8201;" ><!-- thin space, U+2009 ISOpub -->
<!ENTITY zwnj    "&#8204;" ><!-- zero width non-joiner,
                                       U+200C NEW RFC 2070 -->
<!ENTITY zwj     "&#8205;" ><!-- zero width joiner, U+200D NEW RFC 2070 -->
<!ENTITY lrm     "&#8206;" ><!-- left-to-right mark, U+200E NEW RFC 2070 -->
<!ENTITY rlm     "&#8207;" ><!-- right-to-left mark, U+200F NEW RFC 2070 -->
<!ENTITY ndash   "&#8211;" ><!-- en dash, U+2013 ISOpub -->
<!ENTITY mdash   "&#8212;" ><!-- em dash, U+2014 ISOpub -->
<!ENTITY lsquo   "&#8216;" ><!-- left single quotation mark,
                                       U+2018 ISOnum -->
<!ENTITY rsquo   "&#8217;" ><!-- right single quotation mark,
                                       U+2019 ISOnum -->
<!ENTITY sbquo   "&#8218;" ><!-- single low-9 quotation mark, U+201A NEW -->
<!ENTITY ldquo   "&#8220;" ><!-- left double quotation mark,
                                       U+201C ISOnum -->
<!ENTITY rdquo   "&#8221;" ><!-- right double quotation mark,
                                       U+201D ISOnum -->
<!ENTITY bdquo   "&#8222;" ><!-- double low-9 quotation mark, U+201E NEW -->
<!ENTITY dagger  "&#8224;" ><!-- dagger, U+2020 ISOpub -->
<!ENTITY Dagger  "&#8225;" ><!-- double dagger, U+2021 ISOpub -->
<!ENTITY permil  "&#8240;" ><!-- per mille sign, U+2030 ISOtech -->

<!-- lsaquo is proposed but not yet ISO standardized -->
<!ENTITY lsaquo  "&#8249;" ><!-- single left-pointing angle quotation mark,
                                       U+2039 ISO proposed -->
<!-- rsaquo is proposed but not yet ISO standardized -->
```

```
<!ENTITY rsaquo  "&#8250;" ><!-- single right-pointing angle quotation mark,
                                       U+203A ISO proposed -->
<!ENTITY euro    "&#8364;" ><!-- euro sign, U+20AC NEW -->

<!-- end of xhtml-special.ent -->
```

F.1.3. XHTML Mathematical, Greek, and Symbolic Characters

```
<!-- ...................................................................... -->
<!-- ISO Math, Greek and Symbolic Character Entity Set for XHTML .......... -->
<!-- file: xhtml-lat1.ent

     Typical invocation:

       <!ENTITY % xhtml-symbol
           PUBLIC "-//W3C//ENTITIES Symbols for XHTML//EN"
                  "xhtml-symbol.ent" >
       %xhtml-symbol;

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

       PUBLIC "-//W3C//ENTITIES Symbols for XHTML//EN"
       SYSTEM "xhtml-symbol.ent"

     Revision:  $Id: xhtml-symbol.ent,v 1.2 2000/02/03 15:59:28 ahby Exp $ SMI

     Portions (C) International Organization for Standardization 1986:
     Permission to copy in any form is granted for use with conforming
     SGML systems and applications as defined in ISO 8879, provided
     this notice is included in all copies.
-->

<!-- Relevant ISO entity set is given unless names are newly introduced.
     New names (i.e., not in ISO 8879 list) do not clash with any
     existing ISO 8879 entity names. ISO 10646 character numbers
     are given for each character, in hex. CDATA values are decimal
     conversions of the ISO 10646 values and refer to the document
     character set. Names are Unicode 2.0 names.
-->

<!-- Latin Extended-B -->
<!ENTITY fnof     "&#402;" ><!-- latin small f with hook = function
                              = florin, U+0192 ISOtech -->

<!-- Greek -->
<!ENTITY Alpha    "&#913;" ><!-- greek capital letter alpha, U+0391 -->
<!ENTITY Beta     "&#914;" ><!-- greek capital letter beta, U+0392 -->
<!ENTITY Gamma    "&#915;" ><!-- greek capital letter gamma, U+0393 ISOgrk3 -->
<!ENTITY Delta    "&#916;" ><!-- greek capital letter delta, U+0394 ISOgrk3 -->
<!ENTITY Epsilon  "&#917;" ><!-- greek capital letter epsilon, U+0395 -->
<!ENTITY Zeta     "&#918;" ><!-- greek capital letter zeta, U+0396 -->
<!ENTITY Eta      "&#919;" ><!-- greek capital letter eta, U+0397 -->
<!ENTITY Theta    "&#920;" ><!-- greek capital letter theta, U+0398 ISOgrk3 -->
<!ENTITY Iota     "&#921;" ><!-- greek capital letter iota, U+0399 -->
<!ENTITY Kappa    "&#922;" ><!-- greek capital letter kappa, U+039A -->
<!ENTITY Lambda   "&#923;" ><!-- greek capital letter lambda, U+039B ISOgrk3 -->
```

```
<!ENTITY Mu       "&#924;" ><!-- greek capital letter mu, U+039C -->
<!ENTITY Nu       "&#925;" ><!-- greek capital letter nu, U+039D -->
<!ENTITY Xi       "&#926;" ><!-- greek capital letter xi, U+039E ISOgrk3 -->
<!ENTITY Omicron  "&#927;" ><!-- greek capital letter omicron, U+039F -->
<!ENTITY Pi       "&#928;" ><!-- greek capital letter pi, U+03A0 ISOgrk3 -->
<!ENTITY Rho      "&#929;" ><!-- greek capital letter rho, U+03A1 -->
<!-- there is no Sigmaf, and no U+03A2 character either -->
<!ENTITY Sigma    "&#931;" ><!-- greek capital letter sigma, U+03A3 ISOgrk3 -->
<!ENTITY Tau      "&#932;" ><!-- greek capital letter tau, U+03A4 -->
<!ENTITY Upsilon  "&#933;" ><!-- greek capital letter upsilon,
                              U+03A5 ISOgrk3 -->
<!ENTITY Phi      "&#934;" ><!-- greek capital letter phi, U+03A6 ISOgrk3 -->
<!ENTITY Chi      "&#935;" ><!-- greek capital letter chi, U+03A7 -->
<!ENTITY Psi      "&#936;" ><!-- greek capital letter psi, U+03A8 ISOgrk3 -->
<!ENTITY Omega    "&#937;" ><!-- greek capital letter omega, U+03A9 ISOgrk3 -->
<!ENTITY alpha    "&#945;" ><!-- greek small letter alpha, U+03B1 ISOgrk3 -->
<!ENTITY beta     "&#946;" ><!-- greek small letter beta, U+03B2 ISOgrk3 -->
<!ENTITY gamma    "&#947;" ><!-- greek small letter gamma, U+03B3 ISOgrk3 -->
<!ENTITY delta    "&#948;" ><!-- greek small letter delta, U+03B4 ISOgrk3 -->
<!ENTITY epsilon  "&#949;" ><!-- greek small letter epsilon, U+03B5 ISOgrk3 -->
<!ENTITY zeta     "&#950;" ><!-- greek small letter zeta, U+03B6 ISOgrk3 -->
<!ENTITY eta      "&#951;" ><!-- greek small letter eta, U+03B7 ISOgrk3 -->
<!ENTITY theta    "&#952;" ><!-- greek small letter theta, U+03B8 ISOgrk3 -->
<!ENTITY iota     "&#953;" ><!-- greek small letter iota, U+03B9 ISOgrk3 -->
<!ENTITY kappa    "&#954;" ><!-- greek small letter kappa, U+03BA ISOgrk3 -->
<!ENTITY lambda   "&#955;" ><!-- greek small letter lambda, U+03BB ISOgrk3 -->
<!ENTITY mu       "&#956;" ><!-- greek small letter mu, U+03BC ISOgrk3 -->
<!ENTITY nu       "&#957;" ><!-- greek small letter nu, U+03BD ISOgrk3 -->
<!ENTITY xi       "&#958;" ><!-- greek small letter xi, U+03BE ISOgrk3 -->
<!ENTITY omicron  "&#959;" ><!-- greek small letter omicron, U+03BF NEW -->
<!ENTITY pi       "&#960;" ><!-- greek small letter pi, U+03C0 ISOgrk3 -->
<!ENTITY rho      "&#961;" ><!-- greek small letter rho, U+03C1 ISOgrk3 -->
<!ENTITY sigmaf   "&#962;" ><!-- greek small letter final sigma,
                              U+03C2 ISOgrk3 -->
<!ENTITY sigma    "&#963;" ><!-- greek small letter sigma, U+03C3 ISOgrk3 -->
<!ENTITY tau      "&#964;" ><!-- greek small letter tau, U+03C4 ISOgrk3 -->
<!ENTITY upsilon  "&#965;" ><!-- greek small letter upsilon,
                              U+03C5 ISOgrk3 -->
<!ENTITY phi      "&#966;" ><!-- greek small letter phi, U+03C6 ISOgrk3 -->
<!ENTITY chi      "&#967;" ><!-- greek small letter chi, U+03C7 ISOgrk3 -->
<!ENTITY psi      "&#968;" ><!-- greek small letter psi, U+03C8 ISOgrk3 -->
<!ENTITY omega    "&#969;" ><!-- greek small letter omega, U+03C9 ISOgrk3 -->
<!ENTITY thetasym "&#977;" ><!-- greek small letter theta symbol,
                              U+03D1 NEW -->
<!ENTITY upsih    "&#978;" ><!-- greek upsilon with hook symbol,
                              U+03D2 NEW -->
<!ENTITY piv      "&#982;" ><!-- greek pi symbol, U+03D6 ISOgrk3 -->

<!-- General Punctuation -->
<!ENTITY bull     "&#8226;" ><!-- bullet = black small circle,
                               U+2022 ISOpub  -->
<!-- bullet is NOT the same as bullet operator, U+2219 -->
<!ENTITY hellip   "&#8230;" ><!-- horizontal ellipsis = three dot leader,
                               U+2026 ISOpub  -->
<!ENTITY prime    "&#8242;" ><!-- prime = minutes = feet, U+2032 ISOtech -->
<!ENTITY Prime    "&#8243;" ><!-- double prime = seconds = inches,
                               U+2033 ISOtech -->
```

```
<!ENTITY oline    "&#8254;" ><!-- overline = spacing overscore,
                              U+203E NEW -->
<!ENTITY frasl    "&#8260;" ><!-- fraction slash, U+2044 NEW -->

<!-- Letterlike Symbols -->
<!ENTITY weierp   "&#8472;" ><!-- script capital P = power set
                              = Weierstrass p, U+2118 ISOamso -->
<!ENTITY image    "&#8465;" ><!-- blackletter capital I = imaginary part,
                              U+2111 ISOamso -->
<!ENTITY real     "&#8476;" ><!-- blackletter capital R = real part symbol,
                              U+211C ISOamso -->
<!ENTITY trade    "&#8482;" ><!-- trade mark sign, U+2122 ISOnum -->
<!ENTITY alefsym  "&#8501;" ><!-- alef symbol = first transfinite cardinal,
                              U+2135 NEW -->
<!-- alef symbol is NOT the same as hebrew letter alef,
     U+05D0 although the same glyph could be used to depict both characters -->

<!-- Arrows -->
<!ENTITY larr     "&#8592;" ><!-- leftwards arrow, U+2190 ISOnum -->
<!ENTITY uarr     "&#8593;" ><!-- upwards arrow, U+2191 ISOnum-->
<!ENTITY rarr     "&#8594;" ><!-- rightwards arrow, U+2192 ISOnum -->
<!ENTITY darr     "&#8595;" ><!-- downwards arrow, U+2193 ISOnum -->
<!ENTITY harr     "&#8596;" ><!-- left right arrow, U+2194 ISOamsa -->
<!ENTITY crarr    "&#8629;" ><!-- downwards arrow with corner leftwards
                              = carriage return, U+21B5 NEW -->
<!ENTITY lArr     "&#8656;" ><!-- leftwards double arrow, U+21D0 ISOtech -->
<!-- Unicode does not say that lArr is the same as the ’is implied by’ arrow
    but also does not have any other character for that function. So ? lArr can
    be used for ’is implied by’ as ISOtech suggests -->
<!ENTITY uArr     "&#8657;" ><!-- upwards double arrow, U+21D1 ISOamsa -->
<!ENTITY rArr     "&#8658;" ><!-- rightwards double arrow,
                              U+21D2 ISOtech -->
<!-- Unicode does not say this is the ’implies’ character but does not have
     another character with this function so ?
     rArr can be used for ’implies’ as ISOtech suggests -->
<!ENTITY dArr     "&#8659;" ><!-- downwards double arrow, U+21D3 ISOamsa -->
<!ENTITY hArr     "&#8660;" ><!-- left right double arrow,
                              U+21D4 ISOamsa -->

<!-- Mathematical Operators -->
<!ENTITY forall   "&#8704;" ><!-- for all, U+2200 ISOtech -->
<!ENTITY part     "&#8706;" ><!-- partial differential, U+2202 ISOtech  -->
<!ENTITY exist    "&#8707;" ><!-- there exists, U+2203 ISOtech -->
<!ENTITY empty    "&#8709;" ><!-- empty set = null set = diameter,
                              U+2205 ISOamso -->
<!ENTITY nabla    "&#8711;" ><!-- nabla = backward difference,
                              U+2207 ISOtech -->
<!ENTITY isin     "&#8712;" ><!-- element of, U+2208 ISOtech -->
<!ENTITY notin    "&#8713;" ><!-- not an element of, U+2209 ISOtech -->
<!ENTITY ni       "&#8715;" ><!-- contains as member, U+220B ISOtech -->
<!-- should there be a more memorable name than ’ni’? -->
<!ENTITY prod     "&#8719;" ><!-- n-ary product = product sign,
                              U+220F ISOamsb -->
<!-- prod is NOT the same character as U+03A0 ’greek capital letter pi’ though
     the same glyph might be used for both -->
<!ENTITY sum      "&#8721;" ><!-- n-ary sumation, U+2211 ISOamsb -->
<!-- sum is NOT the same character as U+03A3 ’greek capital letter sigma’
```

```
     though the same glyph might be used for both -->
<!ENTITY minus    "&#8722;" ><!-- minus sign, U+2212 ISOtech -->
<!ENTITY lowast   "&#8727;" ><!-- asterisk operator, U+2217 ISOtech -->
<!ENTITY radic    "&#8730;" ><!-- square root = radical sign,
                                 U+221A ISOtech -->
<!ENTITY prop     "&#8733;" ><!-- proportional to, U+221D ISOtech -->
<!ENTITY infin    "&#8734;" ><!-- infinity, U+221E ISOtech -->
<!ENTITY ang      "&#8736;" ><!-- angle, U+2220 ISOamso -->
<!ENTITY and      "&#8743;" ><!-- logical and = wedge, U+2227 ISOtech -->
<!ENTITY or       "&#8744;" ><!-- logical or = vee, U+2228 ISOtech -->
<!ENTITY cap      "&#8745;" ><!-- intersection = cap, U+2229 ISOtech -->
<!ENTITY cup      "&#8746;" ><!-- union = cup, U+222A ISOtech -->
<!ENTITY int      "&#8747;" ><!-- integral, U+222B ISOtech -->
<!ENTITY there4   "&#8756;" ><!-- therefore, U+2234 ISOtech -->
<!ENTITY sim      "&#8764;" ><!-- tilde operator = varies with = similar to,
                                 U+223C ISOtech -->
<!-- tilde operator is NOT the same character as the tilde, U+007E,
     although the same glyph might be used to represent both  -->
<!ENTITY cong     "&#8773;" ><!-- approximately equal to, U+2245 ISOtech -->
<!ENTITY asymp    "&#8776;" ><!-- almost equal to = asymptotic to,
                                 U+2248 ISOamsr -->
<!ENTITY ne       "&#8800;" ><!-- not equal to, U+2260 ISOtech -->
<!ENTITY equiv    "&#8801;" ><!-- identical to, U+2261 ISOtech -->
<!ENTITY le       "&#8804;" ><!-- less-than or equal to, U+2264 ISOtech -->
<!ENTITY ge       "&#8805;" ><!-- greater-than or equal to,
                                 U+2265 ISOtech -->
<!ENTITY sub      "&#8834;" ><!-- subset of, U+2282 ISOtech -->
<!ENTITY sup      "&#8835;" ><!-- superset of, U+2283 ISOtech -->
<!-- note that nsup, ’not a superset of, U+2283’ is not covered by the Symbol
     font encoding and is not included. Should it be, for symmetry?
     It is in ISOamsn  -->
<!ENTITY nsub     "&#8836;" ><!-- not a subset of, U+2284 ISOamsn -->
<!ENTITY sube     "&#8838;" ><!-- subset of or equal to, U+2286 ISOtech -->
<!ENTITY supe     "&#8839;" ><!-- superset of or equal to,
                                 U+2287 ISOtech -->
<!ENTITY oplus    "&#8853;" ><!-- circled plus = direct sum,
                                 U+2295 ISOamsb -->
<!ENTITY otimes   "&#8855;" ><!-- circled times = vector product,
                                 U+2297 ISOamsb -->
<!ENTITY perp     "&#8869;" ><!-- up tack = orthogonal to = perpendicular,
                                 U+22A5 ISOtech -->
<!ENTITY sdot     "&#8901;" ><!-- dot operator, U+22C5 ISOamsb -->
<!-- dot operator is NOT the same character as U+00B7 middle dot -->

<!-- Miscellaneous Technical -->
<!ENTITY lceil    "&#8968;" ><!-- left ceiling = apl upstile,
                                 U+2308 ISOamsc  -->
<!ENTITY rceil    "&#8969;" ><!-- right ceiling, U+2309 ISOamsc  -->
<!ENTITY lfloor   "&#8970;" ><!-- left floor = apl downstile,
                                 U+230A ISOamsc  -->
<!ENTITY rfloor   "&#8971;" ><!-- right floor, U+230B ISOamsc  -->
<!ENTITY lang     "&#9001;" ><!-- left-pointing angle bracket = bra,
                                 U+2329 ISOtech -->
<!-- lang is NOT the same character as U+003C ’less than’
     or U+2039 ’single left-pointing angle quotation mark’ -->
<!ENTITY rang     "&#9002;" ><!-- right-pointing angle bracket = ket,
                                 U+232A ISOtech -->
```

```
<!-- rang is NOT the same character as U+003E ’greater than’
     or U+203A ’single right-pointing angle quotation mark’ -->

<!-- Geometric Shapes -->
<!ENTITY loz      "&#9674;" ><!-- lozenge, U+25CA ISOpub -->

<!-- Miscellaneous Symbols -->
<!ENTITY spades   "&#9824;" ><!-- black spade suit, U+2660 ISOpub -->
<!-- black here seems to mean filled as opposed to hollow -->
<!ENTITY clubs    "&#9827;" ><!-- black club suit = shamrock,
                               U+2663 ISOpub -->
<!ENTITY hearts   "&#9829;" ><!-- black heart suit = valentine,
                               U+2665 ISOpub -->
<!ENTITY diams    "&#9830;" ><!-- black diamond suit, U+2666 ISOpub -->

<!-- end of xhtml-symbol.ent -->
```

F.2. 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:

```
<!-- ...................................................................... -->
<!-- XHTML Modular Framework Module  ...................................... -->
<!-- file: xhtml-framework-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-framework-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ENTITIES XHTML Modular Framework 1.0//EN"
       SYSTEM "xhtml-framework-1.mod"

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

<!-- Modular Framework

     This required module instantiates the modules needed
     to support the XHTML modularization model, including:

        +  notations
        +  datatypes
        +  namespace-qualified names
        +  common attributes
        +  document model
        +  character entities

     The Intrinsic Events module is ignored by default but
     occurs in this module because it must be instantiated
     prior to Attributes but after Datatypes.
```

```
-->

<!ENTITY % xhtml-arch.module "INCLUDE" >
<![%xhtml-arch.module;[
<!ENTITY % xhtml-arch.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Base Architecture 1.0//EN"
            "xhtml-arch-1.mod" >
%xhtml-arch.mod;]]>

<!ENTITY % xhtml-notations.module "INCLUDE" >
<![%xhtml-notations.module;[
<!ENTITY % xhtml-notations.mod
     PUBLIC "-//W3C//NOTATIONS XHTML Notations 1.0//EN"
            "xhtml-notations-1.mod" >
%xhtml-notations.mod;]]>

<!ENTITY % xhtml-datatypes.module "INCLUDE" >
<![%xhtml-datatypes.module;[
<!ENTITY % xhtml-datatypes.mod
     PUBLIC "-//W3C//ENTITIES XHTML Datatypes 1.0//EN"
            "xhtml-datatypes-1.mod" >
%xhtml-datatypes.mod;]]>

<!-- placeholder for XLink support module -->
<!ENTITY % xhtml-xlink.mod "" >
%xhtml-xlink.mod;

<!ENTITY % xhtml-qname.module "INCLUDE" >
<![%xhtml-qname.module;[
<!ENTITY % xhtml-qname.mod
     PUBLIC "-//W3C//ENTITIES XHTML Qualified Names 1.0//EN"
            "xhtml-qname-1.mod" >
%xhtml-qname.mod;]]>

<!ENTITY % xhtml-events.module "IGNORE" >
<![%xhtml-events.module;[
<!ENTITY % xhtml-events.mod
     PUBLIC "-//W3C//ENTITIES XHTML Intrinsic Events 1.0//EN"
            "xhtml-events-1.mod" >
%xhtml-events.mod;]]>

<!ENTITY % xhtml-attribs.module "INCLUDE" >
<![%xhtml-attribs.module;[
<!ENTITY % xhtml-attribs.mod
     PUBLIC "-//W3C//ENTITIES XHTML Common Attributes 1.0//EN"
            "xhtml-attribs-1.mod" >
%xhtml-attribs.mod;]]>

<!-- placeholder for content model redeclarations -->
<!ENTITY % xhtml-model.redecl "" >
%xhtml-model.redecl;

<!ENTITY % xhtml-model.module "INCLUDE" >
<![%xhtml-model.module;[
<!-- instantiate the Document Model module declared in the DTD driver
-->
%xhtml-model.mod;]]>
```

```
<!ENTITY % xhtml-charent.module "INCLUDE" >
<![%xhtml-charent.module;[
<!ENTITY % xhtml-charent.mod
     PUBLIC "-//W3C//ENTITIES XHTML Character Entities 1.0//EN"
            "xhtml-charent-1.mod" >
%xhtml-charent.mod;]]>

<!-- end of xhtml-framework-1.mod -->
```

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. The Modular framework also relies upon the following component modules:

F.2.1. XHTML Base Architecture

```
<!-- ...................................................................... -->
<!-- XHTML Base Architecture Module  ...................................... -->
<!-- file: xhtml-arch-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-arch-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Base Architecture 1.0//EN"
       SYSTEM "xhtml-arch-1.mod"

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

<!-- This optional module includes declarations that enable XHTML to be used
     as a base architecture according to the ’Architectural Forms Definition
     Requirements’ (Annex A.3, ISO/IEC 10744, 2nd edition). For more information
     on use of architectural forms, see the HyTime web site at:

         http://www.hytime.org/
-->

<?IS10744 ArcBase xhtml ?>

<!NOTATION xhtml PUBLIC "-//W3C//NOTATION AFDR ARCBASE XHTML 1.1//EN" >

<!-- Entity declaration for associated Architectural DTD
-->
<!ENTITY xhtml-arch.dtd
      PUBLIC "-//W3C//DTD XHTML Architecture 1.1//EN"
             "xhtml11-arch.dtd" >

<?IS10744:arch xhtml
    public-id       =  "-//W3C//NOTATION AFDR ARCBASE XHTML 1.1//EN"
    dtd-public-id   =  "-//W3C//DTD XHTML 1.1//EN"
    dtd-system-id   =  "xhtml11.dtd"
    doc-elem-form   =  "html"
    form-att        =  "html"
    renamer-att     =  "htnames"
    suppressor-att  =  "htsupp"
    data-ignore-att =  "htign"
    auto            =  "ArcAuto"
    options         =  "HtModReq HtModOpt"
    HtModReq        =  "Framework Text Hypertext Lists Structure"
    HtModOpt        =  "Standard"
?>

<!-- end of xhtml-arch-1.mod -->
```

F.2.2. XHTML Notations

```
Module DTD/xhtml-notation-1.mod not found!
```

F.2.3. XHTML Datatypes

```
<!-- ....................................................................... -->
<!-- XHTML Datatypes Module  .............................................. -->
<!-- file: xhtml-datatypes-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-datatypes-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ENTITIES XHTML Datatypes 1.0//EN"
       SYSTEM "xhtml-datatypes-1.mod"

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

<!-- Datatypes

     defines containers for the following datatypes, many of
     these imported from other specifications and standards.
-->

<!-- Length defined for cellpadding/cellspacing -->

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

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

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

<!-- pixel, percentage, or relative -->
<!ENTITY % MultiLength.datatype "CDATA" >

<!-- one or more digits (NUMBER) -->
```

```
<!ENTITY % Number.datatype "CDATA" >

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

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

<!-- textual content -->
<!ENTITY % Text.datatype "CDATA" >

<!-- Imported Datatypes ................................ -->

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

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

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

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

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

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

<!-- formal public identifier, as per [ISO8879] -->
<!ENTITY % FPI.datatype "CDATA" >

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

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

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

<!-- end of xhtml-datatypes-1.mod -->
```

F.2.4. XHTML Common Attribute Definitions

```
<!-- ...................................................................... -->
<!-- XHTML Common Attributes Module  ...................................... -->
<!-- file: xhtml-attribs-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-attribs-1.mod,v 1.12 2000/09/27 18:06:43 radams Exp $ SMI

     This DTD module is identified by the PUBLIC and SYSTEM identifiers:
       PUBLIC "-//W3C//ENTITIES XHTML Common Attributes 1.0//EN"
       SYSTEM "xhtml-attribs-1.mod"

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

<!-- Common Attributes

     This module declares many of the common attributes for the XHTML DTD.
     %NS.decl.attrib; is declared in the XHTML Qname module.
-->

<!ENTITY % id.attrib
     "id           ID                        #IMPLIED"
>

<!ENTITY % class.attrib
     "class        NMTOKENS                  #IMPLIED"
>

<!ENTITY % title.attrib
     "title        %Text.datatype;           #IMPLIED"
>

<!ENTITY % Core.extra.attrib "" >

<!ENTITY % Core.attrib
     "%XHTML.xmlns.attrib;
      %id.attrib;
      %class.attrib;
      %title.attrib;
      %Core.extra.attrib;"
>

<!ENTITY % lang.attrib
     "xml:lang     %LanguageCode.datatype;  #IMPLIED"
>

<![%XHTML.bidi;[
<!ENTITY % dir.attrib
     "dir          ( ltr | rtl )            #IMPLIED"
>

<!ENTITY % I18n.attrib
     "%dir.attrib;
      %lang.attrib;"
>

]]>
<!ENTITY % I18n.attrib
     "%lang.attrib;"
>

<!ENTITY % Common.extra.attrib "" >
```

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

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

<!-- end of xhtml-attribs-1.mod -->
```

F.2.5. XHTML Qualified Names

```
<!-- ...................................................................... -->
<!-- XHTML Qname Module  ................................................... -->
<!-- file: xhtml-qname-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-qname-1.mod,v 1.13 2000/09/28 21:35:41 radams Exp $ SMI

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

       PUBLIC "-//W3C//ENTITIES XHTML Qualified Names 1.0//EN"
       SYSTEM "xhtml-qname-1.mod"

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

<!-- XHTML Qname (Qualified Name) Module

     This module is contained in two parts, labeled Section 'A' and 'B':

       Section A declares parameter entities to support namespace-
       qualified names, namespace declarations, and name prefixing
       for XHTML and extensions.

       Section B declares parameter entities used to provide
       namespace-qualified names for all XHTML element types:

         %applet.qname;   the xmlns-qualified name for <applet>
         %base.qname;     the xmlns-qualified name for <base>
         ...

     XHTML extensions would create a module similar to this one.
     Included in the XHTML distribution is a template module
     ('template-qname-1.mod') suitable for this purpose.
-->

<!-- Section A: XHTML XML Namespace Framework :::::::::::::::::::: -->

<!-- 1. Declare a %XHTML.prefixed; conditional section keyword, used
        to activate namespace prefixing. The default value should
        inherit '%NS.prefixed;' from the DTD driver, so that unless
        overridden, the default behaviour follows the overall DTD
        prefixing scheme.
-->
<!ENTITY % NS.prefixed "IGNORE" >
<!ENTITY % XHTML.prefixed "%NS.prefixed;" >

<!-- 2. Declare a parameter entity (eg., %XHTML.xmlns;) containing
        the URI reference used to identify the XHTML namespace:
-->
<!ENTITY % XHTML.xmlns  "http://www.w3.org/1999/xhtml" >

<!-- 3. Declare parameter entities (eg., %XHTML.prefix;) containing
        the default namespace prefix string(s) to use when prefixing
        is enabled. This may be overridden in the DTD driver or the
        internal subset of an document instance. If no default prefix
        is desired, this may be declared as an empty string.

     NOTE: As specified in [XMLNAMES], the namespace prefix serves
     as a proxy for the URI reference, and is not in itself significant.
-->
<!ENTITY % XHTML.prefix  "" >

<!-- 4. Declare parameter entities (eg., %XHTML.pfx;) containing the
        colonized prefix(es) (eg., '%XHTML.prefix;:') used when
        prefixing is active, an empty string when it is not.
-->
<![%XHTML.prefixed;[
<!ENTITY % XHTML.pfx  "%XHTML.prefix;:" >
]]>
<!ENTITY % XHTML.pfx  "" >

<!-- declare qualified name extensions here ............ -->
<!ENTITY % xhtml-qname-extra.mod "" >
%xhtml-qname-extra.mod;

<!-- 5. The parameter entity %XHTML.xmlns.extra.attrib; may be
        redeclared to contain any non-XHTML namespace declaration
        attributes for namespaces embedded in XHTML. The default
        is an empty string.  XLink should be included here if used
        in the DTD.
-->
<!ENTITY % XHTML.xmlns.extra.attrib "" >

<!-- The remainder of Section A is only followed in XHTML, not extensions. -->

<!-- Declare a parameter entity %NS.decl.attrib; containing
     all XML Namespace declarations used in the DTD, plus the
     xmlns declaration for XHTML, its form dependent on whether
     prefixing is active.
-->
<![%XHTML.prefixed;[
<!ENTITY % NS.decl.attrib
     "xmlns:%XHTML.prefix;  %URI.datatype;   #FIXED '%XHTML.xmlns;'
      %XHTML.xmlns.extra.attrib;"
>
]]>
```

```
<!ENTITY % NS.decl.attrib
     "%XHTML.xmlns.extra.attrib;"
>

<!-- This is a placeholder for future XLink support.
-->
<!ENTITY % XLINK.xmlns.attrib "" >

<!-- Declare a parameter entity %NS.decl.attrib; containing all
     XML namespace declaration attributes used by XHTML, including
     a default xmlns attribute when prefixing is inactive.
-->
<![%XHTML.prefixed;[
<!ENTITY % XHTML.xmlns.attrib
     "%NS.decl.attrib;
      %XLINK.xmlns.attrib;"
>
]]>
<!ENTITY % XHTML.xmlns.attrib
     "xmlns        %URI.datatype;           #FIXED '%XHTML.xmlns;'
      %XLINK.xmlns.attrib;"
>

<!-- placeholder for qualified name redeclarations -->
<!ENTITY % xhtml-qname.redecl "" >
%xhtml-qname.redecl;

<!-- Section B: XHTML Qualified Names ::::::::::::::::::::::::::::: -->

<!-- 6. This section declares parameter entities used to provide
        namespace-qualified names for all XHTML element types.
-->

<!-- module:  xhtml-applet-1.mod -->
<!ENTITY % applet.qname  "%XHTML.pfx;applet" >

<!-- module:  xhtml-base-1.mod -->
<!ENTITY % base.qname    "%XHTML.pfx;base" >

<!-- module:  xhtml-bdo-1.mod -->
<!ENTITY % bdo.qname     "%XHTML.pfx;bdo" >

<!-- module:  xhtml-blkphras-1.mod -->
<!ENTITY % address.qname "%XHTML.pfx;address" >
<!ENTITY % blockquote.qname  "%XHTML.pfx;blockquote" >
<!ENTITY % pre.qname     "%XHTML.pfx;pre" >
<!ENTITY % h1.qname      "%XHTML.pfx;h1" >
<!ENTITY % h2.qname      "%XHTML.pfx;h2" >
<!ENTITY % h3.qname      "%XHTML.pfx;h3" >
<!ENTITY % h4.qname      "%XHTML.pfx;h4" >
<!ENTITY % h5.qname      "%XHTML.pfx;h5" >
<!ENTITY % h6.qname      "%XHTML.pfx;h6" >

<!-- module:  xhtml-blkpres-1.mod -->
<!ENTITY % hr.qname      "%XHTML.pfx;hr" >

<!-- module:  xhtml-blkstruct-1.mod -->
```

```
<!ENTITY % div.qname     "%XHTML.pfx;div" >
<!ENTITY % p.qname       "%XHTML.pfx;p" >

<!-- module:  xhtml-edit-1.mod -->
<!ENTITY % ins.qname     "%XHTML.pfx;ins" >
<!ENTITY % del.qname     "%XHTML.pfx;del" >

<!-- module:  xhtml-form-1.mod -->
<!ENTITY % form.qname    "%XHTML.pfx;form" >
<!ENTITY % label.qname   "%XHTML.pfx;label" >
<!ENTITY % input.qname   "%XHTML.pfx;input" >
<!ENTITY % select.qname  "%XHTML.pfx;select" >
<!ENTITY % optgroup.qname  "%XHTML.pfx;optgroup" >
<!ENTITY % option.qname  "%XHTML.pfx;option" >
<!ENTITY % textarea.qname  "%XHTML.pfx;textarea" >
<!ENTITY % fieldset.qname  "%XHTML.pfx;fieldset" >
<!ENTITY % legend.qname  "%XHTML.pfx;legend" >
<!ENTITY % button.qname  "%XHTML.pfx;button" >

<!-- module:  xhtml-hypertext-1.mod -->
<!ENTITY % a.qname       "%XHTML.pfx;a" >

<!-- module:  xhtml-image-1.mod -->
<!ENTITY % img.qname     "%XHTML.pfx;img" >

<!-- module:  xhtml-inlphras-1.mod -->
<!ENTITY % abbr.qname    "%XHTML.pfx;abbr" >
<!ENTITY % acronym.qname "%XHTML.pfx;acronym" >
<!ENTITY % cite.qname    "%XHTML.pfx;cite" >
<!ENTITY % code.qname    "%XHTML.pfx;code" >
<!ENTITY % dfn.qname     "%XHTML.pfx;dfn" >
<!ENTITY % em.qname      "%XHTML.pfx;em" >
<!ENTITY % kbd.qname     "%XHTML.pfx;kbd" >
<!ENTITY % q.qname       "%XHTML.pfx;q" >
<!ENTITY % samp.qname    "%XHTML.pfx;samp" >
<!ENTITY % strong.qname  "%XHTML.pfx;strong" >
<!ENTITY % var.qname     "%XHTML.pfx;var" >

<!-- module:  xhtml-inlpres-1.mod -->
<!ENTITY % b.qname       "%XHTML.pfx;b" >
<!ENTITY % big.qname     "%XHTML.pfx;big" >
<!ENTITY % i.qname       "%XHTML.pfx;i" >
<!ENTITY % small.qname   "%XHTML.pfx;small" >
<!ENTITY % sub.qname     "%XHTML.pfx;sub" >
<!ENTITY % sup.qname     "%XHTML.pfx;sup" >
<!ENTITY % tt.qname      "%XHTML.pfx;tt" >

<!-- module:  xhtml-inlstruct-1.mod -->
<!ENTITY % br.qname      "%XHTML.pfx;br" >
<!ENTITY % span.qname    "%XHTML.pfx;span" >

<!-- module:  xhtml-ismap-1.mod (also csismap, ssismap) -->
<!ENTITY % map.qname     "%XHTML.pfx;map" >
<!ENTITY % area.qname    "%XHTML.pfx;area" >

<!-- module:  xhtml-link-1.mod -->
<!ENTITY % link.qname    "%XHTML.pfx;link" >
```

```
<!-- module:  xhtml-list-1.mod -->
<!ENTITY % dl.qname      "%XHTML.pfx;dl" >
<!ENTITY % dt.qname      "%XHTML.pfx;dt" >
<!ENTITY % dd.qname      "%XHTML.pfx;dd" >
<!ENTITY % ol.qname      "%XHTML.pfx;ol" >
<!ENTITY % ul.qname      "%XHTML.pfx;ul" >
<!ENTITY % li.qname      "%XHTML.pfx;li" >

<!-- module:  xhtml-meta-1.mod -->
<!ENTITY % meta.qname    "%XHTML.pfx;meta" >

<!-- module:  xhtml-param-1.mod -->
<!ENTITY % param.qname   "%XHTML.pfx;param" >

<!-- module:  xhtml-object-1.mod -->
<!ENTITY % object.qname  "%XHTML.pfx;object" >

<!-- module:  xhtml-script-1.mod -->
<!ENTITY % script.qname  "%XHTML.pfx;script" >
<!ENTITY % noscript.qname  "%XHTML.pfx;noscript" >

<!-- module:  xhtml-struct-1.mod -->
<!ENTITY % html.qname    "%XHTML.pfx;html" >
<!ENTITY % head.qname    "%XHTML.pfx;head" >
<!ENTITY % title.qname   "%XHTML.pfx;title" >
<!ENTITY % body.qname    "%XHTML.pfx;body" >

<!-- module:  xhtml-style-1.mod -->
<!ENTITY % style.qname   "%XHTML.pfx;style" >

<!-- module:  xhtml-table-1.mod -->
<!ENTITY % table.qname   "%XHTML.pfx;table" >
<!ENTITY % caption.qname "%XHTML.pfx;caption" >
<!ENTITY % thead.qname   "%XHTML.pfx;thead" >
<!ENTITY % tfoot.qname   "%XHTML.pfx;tfoot" >
<!ENTITY % tbody.qname   "%XHTML.pfx;tbody" >
<!ENTITY % colgroup.qname  "%XHTML.pfx;colgroup" >
<!ENTITY % col.qname     "%XHTML.pfx;col" >
<!ENTITY % tr.qname      "%XHTML.pfx;tr" >
<!ENTITY % th.qname      "%XHTML.pfx;th" >
<!ENTITY % td.qname      "%XHTML.pfx;td" >


<!-- Provisional XHTML 2.0 Qualified Names  ...................... -->

<!-- module:  xhtml-image-2.mod -->
<!ENTITY % alt.qname     "%XHTML.pfx;alt" >

<!-- end of xhtml-qname-1.mod -->
```

F.2.6. XHTML Character Entities

```
<!-- ...................................................................... -->
<!-- XHTML Character Entities Module  ......................................... -->
<!-- file: xhtml-charent-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-charent-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ENTITIES XHTML Character Entities 1.0//EN"
       SYSTEM "xhtml-charent-1.mod"

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

<!-- Character Entities for XHTML

     This module declares the set of character entities for XHTML,
     including the Latin 1, Symbol and Special character collections.
-->

<!ENTITY % xhtml-lat1
    PUBLIC "-//W3C//ENTITIES Latin 1 for XHTML//EN"
           "xhtml-lat1.ent" >
<!ENTITY % xhtml-symbol
    PUBLIC "-//W3C//ENTITIES Symbols for XHTML//EN"
           "xhtml-symbol.ent" >
<!ENTITY % xhtml-special
    PUBLIC "-//W3C//ENTITIES Special for XHTML//EN"
           "xhtml-special.ent" >

%xhtml-lat1;
%xhtml-symbol;
%xhtml-special;

<!-- end of xhtml-charent-1.mod -->
```

F.3. XHTML Module Implementations

This section contains the formal definition of each of the XHTML Abstract Modules as a DTD module.

F.3.1. XHTML Core Modules

F.3.1.1. Structure

```
<!-- ...................................................................... -->
<!-- XHTML Structure Module  .............................................. -->
<!-- file: xhtml-struct-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-struct-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Document Structure 1.0//EN"
       SYSTEM "xhtml-struct-1.mod"

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

<!-- Document Structure

        title, head, body, html

     The Structure Module defines the major structural elements and
     their attributes.

     Note that the content model of the head element type is redeclared
     when the Base Module is included in the DTD.

     The parameter entity containing the XML namespace URI value used
     for XHTML is '%XHTML.xmlns;', defined in the Qualified Names module.
-->

<!-- title: Document Title ............................ -->

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

<!ENTITY % title.element  "INCLUDE" >
<![%title.element;[
<!ENTITY % title.content  "( #PCDATA )" >
<!ENTITY % title.qname  "title" >
<!ELEMENT %title.qname;  %title.content; >
<!-- end of title.element -->]]>

<!ENTITY % title.attlist  "INCLUDE" >
<![%title.attlist;[
<!ATTLIST %title.qname;
      %XHTML.xmlns.attrib;
      %I18n.attrib;
>
<!-- end of title.attlist -->]]>

<!-- head: Document Head .............................. -->
```

```
<!ENTITY % head.element  "INCLUDE" >
<![%head.element;[
<!ENTITY % head.content
    "( %HeadOpts.mix;, %title.qname;, %HeadOpts.mix; )"
>
<!ENTITY % head.qname  "head" >
<!ELEMENT %head.qname;  %head.content; >
<!-- end of head.element -->]]>

<!ENTITY % head.attlist  "INCLUDE" >
<![%head.attlist;[
<!-- reserved for future use with document profiles
-->
<!ENTITY % profile.attrib
     "profile      %URI.datatype;           ’%XHTML.profile;’"
>

<!ATTLIST %head.qname;
      %XHTML.xmlns.attrib;
      %I18n.attrib;
      %profile.attrib;
>
<!-- end of head.attlist -->]]>

<!-- body: Document Body ............................... -->

<!ENTITY % body.element  "INCLUDE" >
<![%body.element;[
<!ENTITY % body.content
     "( %Block.mix; )+"
>
<!ENTITY % body.qname  "body" >
<!ELEMENT %body.qname;  %body.content; >
<!-- end of body.element -->]]>

<!ENTITY % body.attlist  "INCLUDE" >
<![%body.attlist;[
<!ATTLIST %body.qname;
      %Common.attrib;
>
<!-- end of body.attlist -->]]>

<!-- html: XHTML Document Element ...................... -->

<!ENTITY % html.element  "INCLUDE" >
<![%html.element;[
<!ENTITY % html.content  "( %head.qname;, %body.qname; )" >
<!ENTITY % html.qname  "html" >
<!ELEMENT %html.qname;  %html.content; >
<!-- end of html.element -->]]>

<!ENTITY % html.attlist  "INCLUDE" >
<![%html.attlist;[
<!-- version attribute value defined in driver
-->
<!ENTITY % XHTML.version.attrib
     "version      %FPI.datatype;           #FIXED ’%XHTML.version;’"
>

<!-- see the Qualified Names module for information
     on how to extend XHTML using XML namespaces
-->
<!ATTLIST %html.qname;
      %XHTML.xmlns.attrib;
      %XHTML.version.attrib;
      %I18n.attrib;
>
<!-- end of html.attlist -->]]>

<!-- end of xhtml-struct-1.mod -->
```

F.3.1.2. Text

```
<!-- ...................................................................... -->
<!-- XHTML Text Module  ................................................... -->
<!-- file: xhtml-text-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-text-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Text 1.0//EN"
       SYSTEM "xhtml-text-1.mod"

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

<!-- Textual Content

     The Text module includes declarations for all core
     text container elements and their attributes.
-->

<!ENTITY % xhtml-inlstruct.module "INCLUDE" >
<![%xhtml-inlstruct.module;[
<!ENTITY % xhtml-inlstruct.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Inline Structural 1.0//EN"
            "xhtml-inlstruct-1.mod" >
%xhtml-inlstruct.mod;]]>

<!ENTITY % xhtml-inlphras.module "INCLUDE" >
<![%xhtml-inlphras.module;[
<!ENTITY % xhtml-inlphras.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Inline Phrasal 1.0//EN"
            "xhtml-inlphras-1.mod" >
%xhtml-inlphras.mod;]]>

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

```
              "xhtml-blkstruct-1.mod" >
%xhtml-blkstruct.mod;]]>

<!ENTITY % xhtml-blkphras.module "INCLUDE" >
<![%xhtml-blkphras.module;[
<!ENTITY % xhtml-blkphras.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Block Phrasal 1.0//EN"
            "xhtml-blkphras-1.mod" >
%xhtml-blkphras.mod;]]>

<!-- end of xhtml-text-1.mod -->
```

F.3.1.3. Hypertext

```
<!-- ...................................................................... -->
<!-- XHTML Hypertext Module  ............................................... -->
<!-- file: xhtml-hypertext-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-hypertext-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Hypertext 1.0//EN"
       SYSTEM "xhtml-hypertext-1.mod"

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

<!-- Hypertext

        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.
-->

<!-- ............  Anchor Element  ............ -->

<!ENTITY % a.element  "INCLUDE" >
<![%a.element;[
<!ENTITY % a.content
     "( #PCDATA | %InlNoAnchor.mix; )*"
>
<!ENTITY % a.qname  "a" >
<!ELEMENT %a.qname;  %a.content; >
<!-- end of a.element -->]]>

<!ENTITY % a.attlist  "INCLUDE" >
<![%a.attlist;[
<!ATTLIST %a.qname;
      %Common.attrib;
      href         %URI.datatype;           #IMPLIED
      charset      %Charset.datatype;       #IMPLIED
      type         %ContentType.datatype;   #IMPLIED
      hreflang     %LanguageCode.datatype;  #IMPLIED
      rel          %LinkTypes.datatype;     #IMPLIED
      rev          %LinkTypes.datatype;     #IMPLIED
      accesskey    %Character.datatype;     #IMPLIED
      tabindex     %Number.datatype;        #IMPLIED
>
<!-- end of a.attlist -->]]>

<!-- end of xhtml-hypertext-1.mod -->
```

F.3.1.4. Lists

```
<!-- ...................................................................... -->
<!-- XHTML Lists Module  ................................................... -->
<!-- file: xhtml-list-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-list-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Lists 1.0//EN"
       SYSTEM "xhtml-list-1.mod"

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

<!-- Lists

        dl, dt, dd, ol, ul, li

     This module declares the list-oriented element types
     and their attributes.
-->

<!ENTITY % dl.qname  "dl" >
<!ENTITY % dt.qname  "dt" >
<!ENTITY % dd.qname  "dd" >
<!ENTITY % ol.qname  "ol" >
<!ENTITY % ul.qname  "ul" >
<!ENTITY % li.qname  "li" >

<!-- dl: Definition List ............................... -->

<!ENTITY % dl.element  "INCLUDE" >
<![%dl.element;[
<!ENTITY % dl.content  "( %dt.qname; | %dd.qname; )+" >
<!ELEMENT %dl.qname;  %dl.content; >
<!-- end of dl.element -->]]>

<!ENTITY % dl.attlist  "INCLUDE" >
<![%dl.attlist;[
<!ATTLIST %dl.qname;
```

```
      %Common.attrib;
>
<!-- end of dl.attlist -->]]>

<!-- dt: Definition Term ............................... -->

<!ENTITY % dt.element  "INCLUDE" >
<![%dt.element;[
<!ENTITY % dt.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ELEMENT %dt.qname;  %dt.content; >
<!-- end of dt.element -->]]>

<!ENTITY % dt.attlist  "INCLUDE" >
<![%dt.attlist;[
<!ATTLIST %dt.qname;
      %Common.attrib;
>
<!-- end of dt.attlist -->]]>

<!-- dd: Definition Description ........................ -->

<!ENTITY % dd.element  "INCLUDE" >
<![%dd.element;[
<!ENTITY % dd.content
     "( #PCDATA | %Flow.mix; )*"
>
<!ELEMENT %dd.qname;  %dd.content; >
<!-- end of dd.element -->]]>

<!ENTITY % dd.attlist  "INCLUDE" >
<![%dd.attlist;[
<!ATTLIST %dd.qname;
      %Common.attrib;
>
<!-- end of dd.attlist -->]]>

<!-- ol: Ordered List (numbered styles) ................ -->

<!ENTITY % ol.element  "INCLUDE" >
<![%ol.element;[
<!ENTITY % ol.content  "( %li.qname; )+" >
<!ELEMENT %ol.qname;  %ol.content; >
<!-- end of ol.element -->]]>

<!ENTITY % ol.attlist  "INCLUDE" >
<![%ol.attlist;[
<!ATTLIST %ol.qname;
      %Common.attrib;
>
<!-- end of ol.attlist -->]]>

<!-- ul: Unordered List (bullet styles) ................ -->

<!ENTITY % ul.element  "INCLUDE" >
<![%ul.element;[
<!ENTITY % ul.content  "( %li.qname; )+" >
<!ELEMENT %ul.qname;  %ul.content; >
<!-- end of ul.element -->]]>

<!ENTITY % ul.attlist  "INCLUDE" >
<![%ul.attlist;[
<!ATTLIST %ul.qname;
      %Common.attrib;
>
<!-- end of ul.attlist -->]]>

<!-- li: List Item .................................... -->

<!ENTITY % li.element  "INCLUDE" >
<![%li.element;[
<!ENTITY % li.content
     "( #PCDATA | %Flow.mix; )*"
>
<!ELEMENT %li.qname;  %li.content; >
<!-- end of li.element -->]]>

<!ENTITY % li.attlist  "INCLUDE" >
<![%li.attlist;[
<!ATTLIST %li.qname;
      %Common.attrib;
>
<!-- end of li.attlist -->]]>

<!-- end of xhtml-list-1.mod -->
```

F.3.2. Applet

```
<!-- ...................................................................... -->
<!-- XHTML Java Applet Module  ............................................ -->
<!-- file: xhtml-applet-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-applet-1.mod,v 1.9 2000/09/20 14:57:38 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Java Applets 1.0//EN"
       SYSTEM "xhtml-applet-1.mod"

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

<!-- Java Applets

        applet

     This module declares the applet element type and its attributes,
     used to provide support for Java applets. The ’alt’ attribute
     is now required (as it is on images). One of either code or
     object attributes must be present. In the document, place param
     elements before the object elements that require their content.

     Note that use of this module requires instantiation of the
     Param Element Module prior to this module.
-->

<!-- applet: Java Applet .............................. -->

<!ENTITY % applet.element  "INCLUDE" >
<![%applet.element;[
<!ENTITY % applet.content
     "( %param.qname; | %Flow.mix; )*"
>
<!ENTITY % applet.qname  "applet" >
<!ELEMENT %applet.qname;  %applet.content; >
<!-- end of applet.element -->]]>

<!ENTITY % applet.attlist  "INCLUDE" >
<![%applet.attlist;[
<!ATTLIST %applet.qname;
      %Core.attrib;
      alt          %Text.datatype;           #REQUIRED
      archive      CDATA                     #IMPLIED
      code         CDATA                     #IMPLIED
      codebase     %URI.datatype;            #IMPLIED
      object       CDATA                     #IMPLIED
      width        %Length.datatype;         #REQUIRED
      height       %Length.datatype;         #REQUIRED
>
<!-- end of applet.attlist -->]]>

<!-- end of xhtml-applet-1.mod -->
```

F.3.3. Text Modules

F.3.3.1. Presentation

```
<!-- ...................................................................... -->
<!-- XHTML Presentation Module ............................................ -->
<!-- file: xhtml-pres-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-pres-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Presentation 1.0//EN"
       SYSTEM "xhtml-pres-1.mod"

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

<!-- Presentational Elements
```

```
     This module defines elements and their attributes for
     simple presentation-related markup.
-->

<!ENTITY % xhtml-inlpres.module "INCLUDE" >
<![%xhtml-inlpres.module;[
<!ENTITY % xhtml-inlpres.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Inline Presentation 1.0//EN"
            "xhtml-inlpres-1.mod" >
%xhtml-inlpres.mod;]]>

<!ENTITY % xhtml-blkpres.module "INCLUDE" >
<![%xhtml-blkpres.module;[
<!ENTITY % xhtml-blkpres.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Block Presentation 1.0//EN"
            "xhtml-blkpres-1.mod" >
%xhtml-blkpres.mod;]]>

<!-- end of xhtml-pres-1.mod -->
```

F.3.3.2. Edit

```
<!-- ...................................................................... -->
<!-- XHTML Editing Elements Module  ..................................... -->
<!-- file: xhtml-edit-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-edit-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Editing Markup 1.0//EN"
       SYSTEM "xhtml-edit-1.mod"

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

<!-- Editing Elements

        ins, del

     This module declares element types and attributes used to indicate
     inserted and deleted content while editing a document.
-->

<!-- ins: Inserted Text  ............................... -->

<!ENTITY % ins.element  "INCLUDE" >
<![%ins.element;[
<!ENTITY % ins.content
     "( #PCDATA | %Flow.mix; )*"
>
<!ENTITY % ins.qname  "ins" >
<!ELEMENT %ins.qname;  %ins.content; >
<!-- end of ins.element -->]]>

<!ENTITY % ins.attlist  "INCLUDE" >
<![%ins.attlist;[
<!ATTLIST %ins.qname;
      %Common.attrib;
      cite         %URI.datatype;           #IMPLIED
      datetime     %Datetime.datatype;      #IMPLIED
>
<!-- end of ins.attlist -->]]>

<!-- del: Deleted Text  ............................... -->

<!ENTITY % del.element  "INCLUDE" >
<![%del.element;[
<!ENTITY % del.content
     "( #PCDATA | %Flow.mix; )*"
>
<!ENTITY % del.qname  "del" >
<!ELEMENT %del.qname;  %del.content; >
<!-- end of del.element -->]]>

<!ENTITY % del.attlist  "INCLUDE" >
<![%del.attlist;[
<!ATTLIST %del.qname;
      %Common.attrib;
      cite         %URI.datatype;           #IMPLIED
      datetime     %Datetime.datatype;      #IMPLIED
>
<!-- end of del.attlist -->]]>

<!-- end of xhtml-edit-1.mod -->
```

F.3.3.3. Bi-directional Text

```
<!-- ...................................................................... -->
<!-- XHTML BDO Element Module ............................................. -->
<!-- file: xhtml-bdo-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-bdo-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML BDO Element 1.0//EN"
       SYSTEM "xhtml-bdo-1.mod"

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

<!-- Bidirectional Override (bdo) Element

     This modules declares the element ’bdo’, used to override the
     Unicode bidirectional algorithm for selected fragments of text.
     DEPENDENCIES:
     Relies on the conditional section keyword %XHTML.bidi; declared
     as "INCLUDE". Bidirectional text support includes both the bdo
     element and the ’dir’ attribute.
-->

<!ENTITY % bdo.element  "INCLUDE" >
<![%bdo.element;[
<!ENTITY % bdo.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % bdo.qname  "bdo" >
<!ELEMENT %bdo.qname;  %bdo.content; >
<!-- end of bdo.element -->]]>

<!ENTITY % bdo.attlist  "INCLUDE" >
<![%bdo.attlist;[
<!ATTLIST %bdo.qname;
      %Core.attrib;
      xml:lang     %LanguageCode.datatype;  #IMPLIED
      dir          ( ltr | rtl )            #REQUIRED
>
]]>

<!-- end of xhtml-bdo-1.mod -->
```

F.3.4. Forms

F.3.4.1. Basic Forms

```
<!-- ...................................................................... -->
<!-- XHTML Simplified Forms Module  ....................................... -->
<!-- file: xhtml-basic-form-1.mod

     This is XHTML Basic, a proper subset of XHTML.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-basic-form-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Basic Forms 1.0//EN"
       SYSTEM "xhtml-basic-form-1.mod"

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

<!-- Basic Forms

     This forms module is based on the HTML 3.2 forms model, with
     the WAI-requested addition of the label element. While this
     module essentially mimics the content model and attributes of
     HTML 3.2 forms, the element types declared herein also include
     all HTML 4 common attributes.

        form, label, input, select, option, textarea
```

```
-->

<!-- declare qualified element type names:
-->
<!ENTITY % form.qname  "form" >
<!ENTITY % label.qname  "label" >
<!ENTITY % input.qname  "input" >
<!ENTITY % select.qname  "select" >
<!ENTITY % option.qname  "option" >
<!ENTITY % textarea.qname  "textarea" >

<!-- %BlkNoForm.mix; includes all non-form block elements,
     plus %Misc.class;
-->
<!ENTITY % BlkNoForm.mix
     "%Heading.class;
      | %List.class;
      | %BlkStruct.class;
      %BlkPhras.class;
      %BlkPres.class;
      | %table.qname;
      %Block.extra;
      %Misc.class;"
>

<!-- form: Form Element ............................... -->

<!ENTITY % form.element  "INCLUDE" >
<![%form.element;[
<!ENTITY % form.content
     "( %BlkNoForm.mix; )+"
>
<!ELEMENT %form.qname;  %form.content; >
<!-- end of form.element -->]]>

<!ENTITY % form.attlist  "INCLUDE" >
<![%form.attlist;[
<!ATTLIST %form.qname;
      %Common.attrib;
      action       %URI.datatype;            #REQUIRED
      method       ( get | post )            ’get’
      enctype      %ContentType.datatype;   ’application/x-www-form-urlencoded’
>
<!-- end of form.attlist -->]]>

<!-- label: Form Field Label Text ..................... -->

<!ENTITY % label.element  "INCLUDE" >
<![%label.element;[
<!-- Each label must not contain more than ONE field
-->
<!ENTITY % label.content
     "( #PCDATA
      | %input.qname; | %select.qname; | %textarea.qname;
      | %InlStruct.class;
      %InlPhras.class;
      %I18n.class;
      %InlPres.class;
      %InlSpecial.class;
      %Misc.class; )*"
>
<!ELEMENT %label.qname;  %label.content; >
<!-- end of label.element -->]]>

<!ENTITY % label.attlist  "INCLUDE" >
<![%label.attlist;[
<!ATTLIST %label.qname;
      %Common.attrib;
      for          IDREF                    #IMPLIED
      accesskey    %Character.datatype;     #IMPLIED
>
<!-- end of label.attlist -->]]>

<!-- input: Form Control .............................. -->

<!ENTITY % input.element  "INCLUDE" >
<![%input.element;[
<!ENTITY % input.content  "EMPTY" >
<!ELEMENT %input.qname;  %input.content; >
<!-- end of input.element -->]]>

<!-- Basic Forms removes ’image’ and ’file’ input types.
-->
<!ENTITY % input.attlist  "INCLUDE" >
<![%input.attlist;[
<!ENTITY % InputType.class
     "( text | password | checkbox | radio
      | submit | reset | hidden )"
>
<!-- attribute name required for all but submit & reset
-->
<!ATTLIST %input.qname;
      %Common.attrib;
      type         %InputType.class;        ’text’
      name         CDATA                    #IMPLIED
      value        CDATA                    #IMPLIED
      checked      ( checked )              #IMPLIED
      size         CDATA                    #IMPLIED
      maxlength    %Number.datatype;        #IMPLIED
      src          %URI.datatype;           #IMPLIED
      accesskey    %Character.datatype;     #IMPLIED
>
<!-- end of input.attlist -->]]>

<!-- select: Option Selector .......................... -->

<!ENTITY % select.element  "INCLUDE" >
<![%select.element;[
<!ENTITY % select.content  "( %option.qname; )+" >
<!ELEMENT %select.qname;  %select.content; >
<!-- end of select.element -->]]>

<!ENTITY % select.attlist  "INCLUDE" >
<![%select.attlist;[
<!ATTLIST %select.qname;
      %Common.attrib;
      name         CDATA                   #IMPLIED
      size         %Number.datatype;       #IMPLIED
      multiple     ( multiple )            #IMPLIED
>
<!-- end of select.attlist -->]]>

<!-- option: Selectable Choice ........................ -->

<!ENTITY % option.element  "INCLUDE" >
<![%option.element;[
<!ENTITY % option.content  "( #PCDATA )" >
<!ELEMENT %option.qname;  %option.content; >
<!-- end of option.element -->]]>

<!ENTITY % option.attlist  "INCLUDE" >
<![%option.attlist;[
<!ATTLIST %option.qname;
      %Common.attrib;
      selected     ( selected )            #IMPLIED
      value        CDATA                   #IMPLIED
>
<!-- end of option.attlist -->]]>

<!-- textarea: Multi-Line Text Field .................. -->

<!ENTITY % textarea.element  "INCLUDE" >
<![%textarea.element;[
<!ENTITY % textarea.content  "( #PCDATA )" >
<!ELEMENT %textarea.qname;  %textarea.content; >
<!-- end of textarea.element -->]]>

<!ENTITY % textarea.attlist  "INCLUDE" >
<![%textarea.attlist;[
<!ATTLIST %textarea.qname;
      %Common.attrib;
      name         CDATA                   #IMPLIED
      rows         %Number.datatype;       #REQUIRED
      cols         %Number.datatype;       #REQUIRED
      accesskey    %Character.datatype;    #IMPLIED
>
<!-- end of textarea.attlist -->]]>

<!-- end of xhtml-basic-form-1.mod -->
```

F.3.4.2. Forms

```
<!-- ...................................................................... -->
<!-- XHTML Forms Module  .................................................. -->
<!-- file: xhtml-form-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-form-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

     This DTD module is identified by the PUBLIC and SYSTEM identifiers:
       PUBLIC "-//W3C//ELEMENTS XHTML Forms 1.0//EN"
       SYSTEM "xhtml-form-1.mod"

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

<!-- Forms

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

     This module declares markup to provide support for online
     forms, based on the features found in HTML 4.0 forms.
-->

<!-- declare qualified element type names:
-->
<!ENTITY % form.qname  "form" >
<!ENTITY % label.qname  "label" >
<!ENTITY % input.qname  "input" >
<!ENTITY % select.qname  "select" >
<!ENTITY % optgroup.qname  "optgroup" >
<!ENTITY % option.qname  "option" >
<!ENTITY % textarea.qname  "textarea" >
<!ENTITY % fieldset.qname  "fieldset" >
<!ENTITY % legend.qname  "legend" >
<!ENTITY % button.qname  "button" >

<!-- %BlkNoForm.mix; includes all non-form block elements,
     plus %Misc.class;
-->
<!ENTITY % BlkNoForm.mix
     "%Heading.class;
      | %List.class;
      | %BlkStruct.class;
      %BlkPhras.class;
      %BlkPres.class;
      %Table.class;
      %Block.extra;
      %Misc.class;"
>

<!-- form: Form Element ................................ -->

<!ENTITY % form.element  "INCLUDE" >
<![%form.element;[
<!ENTITY % form.content
     "( %BlkNoForm.mix;
      | %fieldset.qname; )+"
>
<!ELEMENT %form.qname;  %form.content; >
<!-- end of form.element -->]]>

<!ENTITY % form.attlist  "INCLUDE" >
<![%form.attlist;[
<!ATTLIST %form.qname;
      %Common.attrib;
      action       %URI.datatype;           #REQUIRED
      method       ( get | post )           ’get’
      enctype      %ContentType.datatype;   ’application/x-www-form-urlencoded’
      accept-charset %Charsets.datatype;    #IMPLIED
      accept       %ContentTypes.datatype;  #IMPLIED
>
<!-- end of form.attlist -->]]>

<!-- label: Form Field Label Text ...................... -->

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

<!ENTITY % label.element  "INCLUDE" >
<![%label.element;[
<!ENTITY % label.content
     "( #PCDATA
      | %input.qname; | %select.qname; | %textarea.qname; | %button.qname;
      | %InlStruct.class;
      %InlPhras.class;
      %I18n.class;
      %InlPres.class;
      %Anchor.class;
      %InlSpecial.class;
      %Inline.extra;
      %Misc.class; )*"
>
<!ELEMENT %label.qname;  %label.content; >
<!-- end of label.element -->]]>

<!ENTITY % label.attlist  "INCLUDE" >
<![%label.attlist;[
<!ATTLIST %label.qname;
      %Common.attrib;
      for          IDREF                    #IMPLIED
      accesskey    %Character.datatype;     #IMPLIED
>
<!-- end of label.attlist -->]]>

<!-- input: Form Control .............................. -->

<!ENTITY % input.element  "INCLUDE" >
<![%input.element;[
<!ENTITY % input.content  "EMPTY" >
<!ELEMENT %input.qname;  %input.content; >
<!-- end of input.element -->]]>

<!ENTITY % input.attlist  "INCLUDE" >
<![%input.attlist;[
<!ENTITY % InputType.class
     "( text | password | checkbox | radio | submit
      | reset | file | hidden | image | button )"
>
<!-- attribute ’name’ required for all but submit & reset
-->
```

```
<!ATTLIST %input.qname;
      %Common.attrib;
      type         %InputType.class;        'text'
      name         CDATA                    #IMPLIED
      value        CDATA                    #IMPLIED
      checked      ( checked )              #IMPLIED
      disabled     ( disabled )             #IMPLIED
      readonly     ( readonly )             #IMPLIED
      size         %Number.datatype;        #IMPLIED
      maxlength    %Number.datatype;        #IMPLIED
      src          %URI.datatype;           #IMPLIED
      alt          CDATA                    #IMPLIED
      tabindex     %Number.datatype;        #IMPLIED
      accesskey    %Character.datatype;     #IMPLIED
      accept       %ContentTypes.datatype;  #IMPLIED
>
<!-- end of input.attlist -->]]>

<!-- select: Option Selector .......................... -->

<!ENTITY % select.element  "INCLUDE" >
<![%select.element;[
<!ENTITY % select.content
     "( %optgroup.qname; | %option.qname; )+"
>
<!ELEMENT %select.qname;  %select.content; >
<!-- end of select.element -->]]>

<!ENTITY % select.attlist  "INCLUDE" >
<![%select.attlist;[
<!ATTLIST %select.qname;
      %Common.attrib;
      name         CDATA                    #IMPLIED
      size         %Number.datatype;        #IMPLIED
      multiple     ( multiple )             #IMPLIED
      disabled     ( disabled )             #IMPLIED
      tabindex     %Number.datatype;        #IMPLIED
>
<!-- end of select.attlist -->]]>

<!-- optgroup: Option Group ........................... -->

<!ENTITY % optgroup.element  "INCLUDE" >
<![%optgroup.element;[
<!ENTITY % optgroup.content  "( %option.qname; )+" >
<!ELEMENT %optgroup.qname;  %optgroup.content; >
<!-- end of optgroup.element -->]]>

<!ENTITY % optgroup.attlist  "INCLUDE" >
<![%optgroup.attlist;[
<!ATTLIST %optgroup.qname;
      %Common.attrib;
      disabled     ( disabled )             #IMPLIED
      label        %Text.datatype;          #REQUIRED
>
<!-- end of optgroup.attlist -->]]>
```

```
<!-- option: Selectable Choice ........................ -->

<!ENTITY % option.element  "INCLUDE" >
<![%option.element;[
<!ENTITY % option.content  "( #PCDATA )" >
<!ELEMENT %option.qname;  %option.content; >
<!-- end of option.element -->]]>

<!ENTITY % option.attlist  "INCLUDE" >
<![%option.attlist;[
<!ATTLIST %option.qname;
      %Common.attrib;
      selected     ( selected )             #IMPLIED
      disabled     ( disabled )             #IMPLIED
      label        %Text.datatype;          #IMPLIED
      value        CDATA                    #IMPLIED
>
<!-- end of option.attlist -->]]>

<!-- textarea: Multi-Line Text Field .................. -->

<!ENTITY % textarea.element  "INCLUDE" >
<![%textarea.element;[
<!ENTITY % textarea.content  "( #PCDATA )" >
<!ELEMENT %textarea.qname;  %textarea.content; >
<!-- end of textarea.element -->]]>

<!ENTITY % textarea.attlist  "INCLUDE" >
<![%textarea.attlist;[
<!ATTLIST %textarea.qname;
      %Common.attrib;
      name         CDATA                    #IMPLIED
      rows         %Number.datatype;        #REQUIRED
      cols         %Number.datatype;        #REQUIRED
      disabled     ( disabled )             #IMPLIED
      readonly     ( readonly )             #IMPLIED
      tabindex     %Number.datatype;        #IMPLIED
      accesskey    %Character.datatype;     #IMPLIED
>
<!-- end of textarea.attlist -->]]>

<!-- fieldset: Form Control Group ..................... -->

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

<!ENTITY % fieldset.element  "INCLUDE" >
<![%fieldset.element;[
<!ENTITY % fieldset.content
     "( #PCDATA | %legend.qname; | %Flow.mix; )*"
>
<!ELEMENT %fieldset.qname;  %fieldset.content; >
<!-- end of fieldset.element -->]]>

<!ENTITY % fieldset.attlist  "INCLUDE" >
<![%fieldset.attlist;[
<!ATTLIST %fieldset.qname;
      %Common.attrib;
>
<!-- end of fieldset.attlist -->]]>

<!-- legend: Fieldset Legend ........................... -->

<!ENTITY % legend.element  "INCLUDE" >
<![%legend.element;[
<!ENTITY % legend.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ELEMENT %legend.qname;  %legend.content; >
<!-- end of legend.element -->]]>

<!ENTITY % legend.attlist  "INCLUDE" >
<![%legend.attlist;[
<!ATTLIST %legend.qname;
      %Common.attrib;
      accesskey    %Character.datatype;     #IMPLIED
>
<!-- end of legend.attlist -->]]>

<!-- button: Push Button ............................... -->

<!ENTITY % button.element  "INCLUDE" >
<![%button.element;[
<!ENTITY % button.content
     "( #PCDATA
      | %BlkNoForm.mix;
      | %InlStruct.class;
      %InlPhras.class;
      %InlPres.class;
      %I18n.class;
      %InlSpecial.class;
      %Inline.extra; )*"
>
<!ELEMENT %button.qname;  %button.content; >
<!-- end of button.element -->]]>

<!ENTITY % button.attlist  "INCLUDE" >
<![%button.attlist;[
<!ATTLIST %button.qname;
      %Common.attrib;
      name         CDATA                    #IMPLIED
      value        CDATA                    #IMPLIED
      type         ( button | submit | reset ) ’submit’
      disabled     ( disabled )             #IMPLIED
      tabindex     %Number.datatype;        #IMPLIED
      accesskey    %Character.datatype;     #IMPLIED
>
<!-- end of button.attlist -->]]>

<!-- end of xhtml-form-1.mod -->
```

F.3.5. Tables

F.3.5.1. Basic Tables

```
<!-- ...................................................................... -->
<!-- XHTML Basic Table Module  ............................................. -->
<!-- file: xhtml-basic-table-1.mod

     This is XHTML Basic, a proper subset of XHTML.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-basic-table-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Basic Tables 1.0//EN"
       SYSTEM "xhtml-basic-table-1.mod"

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

<!-- Basic Tables

        table, caption, tr, th, td

     This table module declares elements and attributes defining
     a table model based fundamentally on features found in the
     widely-deployed HTML 3.2 table model.  While this module
     mimics the content model and table attributes of HTML 3.2
     tables, the element types declared herein also includes all
     HTML 4 common and most of the HTML 4 table attributes.
-->

<!-- declare qualified element type names:
-->
<!ENTITY % table.qname  "table" >
<!ENTITY % caption.qname  "caption" >
<!ENTITY % tr.qname  "tr" >
<!ENTITY % th.qname  "th" >
<!ENTITY % td.qname  "td" >

<!-- horizontal alignment attributes for cell contents
-->
<!ENTITY % CellHAlign.attrib
     "align        ( left
                   | center
                   | right )                #IMPLIED"
>

<!-- vertical alignment attributes for cell contents
-->
<!ENTITY % CellVAlign.attrib
     "valign       ( top
                   | middle
                   | bottom )               #IMPLIED"
>
```

```
<!-- scope is simpler than axes attribute for common tables
-->
<!ENTITY % scope.attrib
     "scope        ( row | col  )           #IMPLIED"
>

<!-- table: Table Element ............................. -->

<!ENTITY % table.element  "INCLUDE" >
<![%table.element;[
<!ENTITY % table.content
     "( %caption.qname;?, %tr.qname;+ )"
>
<!ELEMENT %table.qname;  %table.content; >
<!-- end of table.element -->]]>

<!ENTITY % table.attlist  "INCLUDE" >
<![%table.attlist;[
<!ATTLIST %table.qname;
      %Common.attrib;
      summary      %Text.datatype;          #IMPLIED
>
<!-- end of table.attlist -->]]>

<!-- caption: Table Caption ........................... -->

<!ENTITY % caption.element  "INCLUDE" >
<![%caption.element;[
<!ENTITY % caption.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ELEMENT %caption.qname;  %caption.content; >
<!-- end of caption.element -->]]>

<!ENTITY % caption.attlist  "INCLUDE" >
<![%caption.attlist;[
<!ATTLIST %caption.qname;
      %Common.attrib;
>
<!-- end of caption.attlist -->]]>

<!-- tr: Table Row .................................... -->

<!ENTITY % tr.element  "INCLUDE" >
<![%tr.element;[
<!ENTITY % tr.content  "( %th.qname; | %td.qname; )+" >
<!ELEMENT %tr.qname;  %tr.content; >
<!-- end of tr.element -->]]>

<!ENTITY % tr.attlist  "INCLUDE" >
<![%tr.attlist;[
<!ATTLIST %tr.qname;
      %Common.attrib;
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of tr.attlist -->]]>

<!-- th: Table Header Cell ............................ -->

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

<!ENTITY % th.element  "INCLUDE" >
<![%th.element;[
<!ENTITY % th.content
     "( #PCDATA | %FlowNoTable.mix; )*"
>
<!ELEMENT %th.qname;  %th.content; >
<!-- end of th.element -->]]>

<!ENTITY % th.attlist  "INCLUDE" >
<![%th.attlist;[
<!ATTLIST %th.qname;
      %Common.attrib;
      abbr         %Text.datatype;           #IMPLIED
      axis         CDATA                     #IMPLIED
      headers      IDREFS                    #IMPLIED
      %scope.attrib;
      rowspan      %Number.datatype;         ’1’
      colspan      %Number.datatype;         ’1’
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of th.attlist -->]]>

<!-- td: Table Data Cell .............................. -->

<!ENTITY % td.element  "INCLUDE" >
<![%td.element;[
<!ENTITY % td.content
     "( #PCDATA | %FlowNoTable.mix; )*"
>
<!ELEMENT %td.qname;  %td.content; >
<!-- end of td.element -->]]>

<!ENTITY % td.attlist  "INCLUDE" >
<![%td.attlist;[
<!ATTLIST %td.qname;
      %Common.attrib;
      abbr         %Text.datatype;           #IMPLIED
      axis         CDATA                     #IMPLIED
      headers      IDREFS                    #IMPLIED
      %scope.attrib;
      rowspan      %Number.datatype;         ’1’
      colspan      %Number.datatype;         ’1’
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of td.attlist -->]]>

<!-- end of xhtml-basic-table-1.mod -->
```

F.3.5.2. Tables

```
<!-- ...................................................................... -->
<!-- XHTML Table Module  ................................................... -->
<!-- file: xhtml-table-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-table-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Tables 1.0//EN"
       SYSTEM "xhtml-table-1.mod"

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

<!-- 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.
-->

<!-- declare qualified element type names:
-->
<!ENTITY % table.qname  "table" >
<!ENTITY % caption.qname  "caption" >
<!ENTITY % thead.qname  "thead" >
<!ENTITY % tfoot.qname  "tfoot" >
<!ENTITY % tbody.qname  "tbody" >
<!ENTITY % colgroup.qname  "colgroup" >
<!ENTITY % col.qname  "col" >
<!ENTITY % tr.qname  "tr" >
<!ENTITY % th.qname  "th" >
<!ENTITY % td.qname  "td" >

<!-- The frame attribute specifies which parts of the frame around
     the table should be rendered. The values are not the same as
     CALS to avoid a name clash with the valign attribute.
-->
<!ENTITY % frame.attrib
     "frame        ( void
                   | above
                   | below
                   | hsides
                   | lhs
                   | rhs
                   | vsides
                   | box
                   | border )               #IMPLIED"
>
```

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

     If rules is absent then assume:

       "none" if border is absent or border="0" otherwise "all"
-->
<!ENTITY % rules.attrib
     "rules        ( none
                   | groups
                   | rows
                   | cols
                   | all )                  #IMPLIED"
>

<!-- horizontal alignment attributes for cell contents
-->
<!ENTITY % CellHAlign.attrib
     "align        ( left
                   | center
                   | right
                   | justify
                   | char )                 #IMPLIED
      char         %Character.datatype;     #IMPLIED
      charoff      %Length.datatype;        #IMPLIED"
>

<!-- vertical alignment attribute for cell contents
-->
<!ENTITY % CellVAlign.attrib
     "valign       ( top
                   | middle
                   | bottom
                   | baseline )             #IMPLIED"
>

<!-- scope is simpler than axes attribute for common tables
-->
<!ENTITY % scope.attrib
     "scope        ( row
                   | col
                   | rowgroup
                   | colgroup )             #IMPLIED"
>

<!-- table: Table Element ............................. -->

<!ENTITY % table.element  "INCLUDE" >
<![%table.element;[
<!ENTITY % table.content
     "( %caption.qname;?, ( %col.qname;* | %colgroup.qname;* ),
      (( %thead.qname;?, %tfoot.qname;?, %tbody.qname;+ ) | ( %tr.qname;+ )))"
>
<!ELEMENT %table.qname;  %table.content; >
<!-- end of table.element -->]]>

<!ENTITY % table.attlist  "INCLUDE" >
<![%table.attlist;[
<!ATTLIST %table.qname;
      %Common.attrib;
      summary      %Text.datatype;          #IMPLIED
      width        %Length.datatype;        #IMPLIED
      border       %Pixels.datatype;        #IMPLIED
      %frame.attrib;
      %rules.attrib;
      cellspacing  %Length.datatype;        #IMPLIED
      cellpadding  %Length.datatype;        #IMPLIED
>
<!-- end of table.attlist -->]]>

<!-- caption: Table Caption ............................ -->

<!ENTITY % caption.element  "INCLUDE" >
<![%caption.element;[
<!ENTITY % caption.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ELEMENT %caption.qname;  %caption.content; >
<!-- end of caption.element -->]]>

<!ENTITY % caption.attlist  "INCLUDE" >
<![%caption.attlist;[
<!ATTLIST %caption.qname;
      %Common.attrib;
>
<!-- end of caption.attlist -->]]>

<!-- thead: Table Header ............................... -->

<!-- Use thead to duplicate headers when breaking table
     across page boundaries, or for static headers when
     tbody sections are rendered in scrolling panel.
-->

<!ENTITY % thead.element  "INCLUDE" >
<![%thead.element;[
<!-- end of thead.element -->]]>
<!ENTITY % thead.content  "( %tr.qname; )+" >
<!ELEMENT %thead.qname;  %thead.content; >

<!ENTITY % thead.attlist  "INCLUDE" >
<![%thead.attlist;[
<!ATTLIST %thead.qname;
      %Common.attrib;
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of thead.attlist -->]]>

<!-- tfoot: Table Footer ............................... -->

<!-- Use tfoot to duplicate footers when breaking table
     across page boundaries, or for static footers when
     tbody sections are rendered in scrolling panel.
-->
```

```
<!ENTITY % tfoot.element  "INCLUDE" >
<![%tfoot.element;[
<!ENTITY % tfoot.content  "( %tr.qname; )+" >
<!ELEMENT %tfoot.qname;  %tfoot.content; >
<!-- end of tfoot.element -->]]>

<!ENTITY % tfoot.attlist  "INCLUDE" >
<![%tfoot.attlist;[
<!ATTLIST %tfoot.qname;
      %Common.attrib;
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of tfoot.attlist -->]]>

<!-- tbody: Table Body ................................ -->

<!-- Use multiple tbody sections when rules are needed
     between groups of table rows.
-->

<!ENTITY % tbody.element  "INCLUDE" >
<![%tbody.element;[
<!ENTITY % tbody.content  "( %tr.qname; )+" >
<!ELEMENT %tbody.qname;  %tbody.content; >
<!-- end of tbody.element -->]]>

<!ENTITY % tbody.attlist  "INCLUDE" >
<![%tbody.attlist;[
<!ATTLIST %tbody.qname;
      %Common.attrib;
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of tbody.attlist -->]]>

<!-- colgroup: Table Column Group ..................... -->

<!-- colgroup groups a set of col elements. It allows you
     to group several semantically-related columns together.
-->

<!ENTITY % colgroup.element  "INCLUDE" >
<![%colgroup.element;[
<!ENTITY % colgroup.content  "( %col.qname; )*" >
<!ELEMENT %colgroup.qname;  %colgroup.content; >
<!-- end of colgroup.element -->]]>

<!ENTITY % colgroup.attlist  "INCLUDE" >
<![%colgroup.attlist;[
<!-- end of colgroup.attlist -->]]>
<!ATTLIST %colgroup.qname;
      %Common.attrib;
      span         %Number.datatype;        ’1’
      width        %MultiLength.datatype;   #IMPLIED
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>

<!-- col: Table Column ................................ -->

<!-- col elements define the alignment properties for
     cells in one or more columns.

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

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

     The span attribute causes the attributes of one
     col element to apply to more than one column.
-->

<!ENTITY % col.element  "INCLUDE" >
<![%col.element;[
<!ENTITY % col.content  "EMPTY" >
<!ELEMENT %col.qname;  %col.content; >
<!-- end of col.element -->]]>

<!ENTITY % col.attlist  "INCLUDE" >
<![%col.attlist;[
<!ATTLIST %col.qname;
      %Common.attrib;
      span         %Number.datatype;        ’1’
      width        %MultiLength.datatype;   #IMPLIED
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of col.attlist -->]]>

<!-- tr: Table Row .................................... -->

<!ENTITY % tr.element  "INCLUDE" >
<![%tr.element;[
<!ENTITY % tr.content  "( %th.qname; | %td.qname; )+" >
<!ELEMENT %tr.qname;  %tr.content; >
<!-- end of tr.element -->]]>

<!ENTITY % tr.attlist  "INCLUDE" >
<![%tr.attlist;[
<!ATTLIST %tr.qname;
      %Common.attrib;
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of tr.attlist -->]]>

<!-- th: Table Header Cell ............................ -->

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

```
<!ENTITY % th.element  "INCLUDE" >
<![%th.element;[
<!ENTITY % th.content
     "( #PCDATA | %Flow.mix; )*"
>
<!ELEMENT %th.qname;  %th.content; >
<!-- end of th.element -->]]>

<!ENTITY % th.attlist  "INCLUDE" >
<![%th.attlist;[
<!ATTLIST %th.qname;
      %Common.attrib;
      abbr         %Text.datatype;          #IMPLIED
      axis         CDATA                    #IMPLIED
      headers      IDREFS                   #IMPLIED
      %scope.attrib;
      rowspan      %Number.datatype;        ’1’
      colspan      %Number.datatype;        ’1’
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of th.attlist -->]]>

<!-- td: Table Data Cell ............................... -->

<!ENTITY % td.element  "INCLUDE" >
<![%td.element;[
<!ENTITY % td.content
     "( #PCDATA | %Flow.mix; )*"
>
<!ELEMENT %td.qname;  %td.content; >
<!-- end of td.element -->]]>

<!ENTITY % td.attlist  "INCLUDE" >
<![%td.attlist;[
<!ATTLIST %td.qname;
      %Common.attrib;
      abbr         %Text.datatype;          #IMPLIED
      axis         CDATA                    #IMPLIED
      headers      IDREFS                   #IMPLIED
      %scope.attrib;
      rowspan      %Number.datatype;        ’1’
      colspan      %Number.datatype;        ’1’
      %CellHAlign.attrib;
      %CellVAlign.attrib;
>
<!-- end of td.attlist -->]]>

<!-- end of xhtml-table-1.mod -->
```

F.3.6. Image

```
<!-- ...................................................................... -->
<!-- XHTML Images Module  ................................................. -->
<!-- file: xhtml-image-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Rovision: $Id: xhtml-image-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Images 1.0//EN"
       SYSTEM "xhtml-image-1.mod"

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

<!-- Images

        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.
-->

<!ENTITY % img.element  "INCLUDE" >
<![%img.element;[
<!ENTITY % img.content  "EMPTY" >
<!ENTITY % img.qname  "img" >
<!ELEMENT %img.qname;  %img.content; >
<!-- end of img.element -->]]>

<!ENTITY % img.attlist  "INCLUDE" >
<![%img.attlist;[
<!ATTLIST %img.qname;
      %Common.attrib;
      src          %URI.datatype;           #REQUIRED
      alt          %Text.datatype;          #REQUIRED
      longdesc     %URI.datatype;           #IMPLIED
      height       %Length.datatype;        #IMPLIED
      width        %Length.datatype;        #IMPLIED
>
<!-- end of img.attlist -->]]>

<!-- end of xhtml-image-1.mod -->
```

F.3.7. Client-side Image Map

```
<!-- ...................................................................... -->
<!-- XHTML Client-side Image Map Module  .................................. -->
<!-- file: xhtml-csismap-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-csismap-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Client-side Image Maps 1.0//EN"
       SYSTEM "xhtml-csismap-1.mod"

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

<!-- Client-side Image Maps

        area, map

     This module declares elements and attributes to support client-side
     image maps. This requires that the Image Module (or a module
     declaring the img element type) be included in the DTD.

     These can be placed in the same document or grouped in a
     separate document, although the latter isn’t widely supported
-->

<!ENTITY % area.element  "INCLUDE" >
<![%area.element;[
<!ENTITY % area.content  "EMPTY" >
<!ENTITY % area.qname  "area" >
<!ELEMENT %area.qname;  %area.content; >
<!-- end of area.element -->]]>

<!ENTITY % Shape.datatype "( rect | circle | poly | default )">
<!ENTITY % Coords.datatype "CDATA" >

<!ENTITY % area.attlist  "INCLUDE" >
<![%area.attlist;[
<!ATTLIST %area.qname;
      %Common.attrib;
      href         %URI.datatype;           #IMPLIED
      shape        %Shape.datatype;         ’rect’
      coords       %Coords.datatype;        #IMPLIED
      nohref       ( nohref )               #IMPLIED
      alt          %Text.datatype;          #REQUIRED
      tabindex     %Number.datatype;        #IMPLIED
      accesskey    %Character.datatype;     #IMPLIED
>
<!-- end of area.attlist -->]]>

<!-- modify anchor attribute definition list
```

```
     to allow for client-side image maps
-->
<!ATTLIST %a.qname;
      shape        %Shape.datatype;         ’rect’
      coords       %Coords.datatype;        #IMPLIED
>

<!-- modify img attribute definition list
     to allow for client-side image maps
-->
<!ATTLIST %img.qname;
      usemap       IDREF                    #IMPLIED
>

<!-- modify form input attribute definition list
     to allow for client-side image maps
-->
<!ATTLIST %input.qname;
      usemap       IDREF                    #IMPLIED
>

<!-- modify object attribute definition list
     to allow for client-side image maps
-->
<!ATTLIST %object.qname;
      usemap       IDREF                    #IMPLIED
>

<!-- ’usemap’ points to the ’id’ attribute of a MAP element,
     which must be in the same document; support for external
     document maps was not widely supported in HTML and is
     eliminated in XHTML.

     It is considered an error for the element pointed to by
     a usemap IDREF to occur in anything but a map element.
-->

<!ENTITY % map.element  "INCLUDE" >
<![%map.element;[
<!ENTITY % map.content
     "(( %Block.mix; ) | %area.qname; )+"
>
<!ENTITY % map.qname  "map" >
<!ELEMENT %map.qname;  %map.content; >
<!-- end of map.element -->]]>

<!ENTITY % map.attlist  "INCLUDE" >
<![%map.attlist;[
<!ATTLIST %map.qname;
      %XHTML.xmlns.attrib;
      id           ID                       #REQUIRED
      %class.attrib;
      %title.attrib;
      %Core.extra.attrib;
      %I18n.attrib;
      %Events.attrib;
>
<!-- end of map.attlist -->]]>

<!-- end of xhtml-csismap-1.mod -->
```

F.3.8. Server-side Image Map

```
<!-- ...................................................................... -->
<!-- XHTML Server-side Image Map Module  .................................. -->
<!-- file: xhtml-ssismap-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-ssismap-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Server-side Image Maps 1.0//EN"
       SYSTEM "xhtml-ssismap-1.mod"

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

<!-- Server-side Image Maps

     This adds the 'ismap' attribute to the img element to
     support server-side processing of a user selection.
-->

<!ATTLIST %img.qname;
      ismap        ( ismap )                #IMPLIED
>

<!-- end of xhtml-ssismap-1.mod -->
```

F.3.9. Object

```
<!-- ...................................................................... -->
<!-- XHTML Embedded Object Module  ........................................ -->
<!-- file: xhtml-object-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-object-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Embedded Object 1.0//EN"
       SYSTEM "xhtml-object-1.mod"

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

<!-- Embedded Objects
```

```
        object

     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.
-->

<!-- object: Generic Embedded Object ................... -->

<!ENTITY % object.element  "INCLUDE" >
<![%object.element;[
<!ENTITY % object.content
     "( #PCDATA | %Flow.mix; | %param.qname; )*"
>
<!ENTITY % object.qname  "object" >
<!ELEMENT %object.qname;  %object.content; >
<!-- end of object.element -->]]>

<!ENTITY % object.attlist  "INCLUDE" >
<![%object.attlist;[
<!ATTLIST %object.qname;
      %Common.attrib;
      declare      ( declare )              #IMPLIED
      classid      %URI.datatype;           #IMPLIED
      codebase     %URI.datatype;           #IMPLIED
      data         %URI.datatype;           #IMPLIED
      type         %ContentType.datatype;   #IMPLIED
      codetype     %ContentType.datatype;   #IMPLIED
      archive      %URIs.datatype;          #IMPLIED
      standby      %Text.datatype;          #IMPLIED
      height       %Length.datatype;        #IMPLIED
      width        %Length.datatype;        #IMPLIED
      name         CDATA                    #IMPLIED
      tabindex     %Number.datatype;        #IMPLIED
>
<!-- end of object.attlist -->]]>

<!-- end of xhtml-object-1.mod -->
```

F.3.10. Frames

```
<!-- ...................................................................... -->
<!-- XHTML Frames Module  ................................................. -->
<!-- file: xhtml-frames-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-frames-1.mod,v 1.11 2000/09/27 22:28:18 radams Exp $ SMI

     This DTD module is identified by the PUBLIC and SYSTEM identifiers:
       PUBLIC "-//W3C//ELEMENTS XHTML Frames 1.0//EN"
       SYSTEM "xhtml-frames-1.mod"

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

<!-- Frames

        frameset, frame, noframes

     This module declares frame-related element types and attributes.
-->

<!ENTITY % frameset.qname  "frameset" >
<!ENTITY % frame.qname  "frame" >
<!ENTITY % noframes.qname  "noframes" >

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

<!-- The content model for XHTML documents depends on whether
     the <head> is followed by a <frameset> or <body> element.
-->

<!ENTITY % frameset.content
     "(( %frameset.qname; | %frame.qname; )+, %noframes.qname;? )" >
<!ELEMENT %frameset.qname;  %frameset.content; >
<!ATTLIST %frameset.qname;
      %Core.attrib;
      rows         %MultiLengths.datatype;  #IMPLIED
      cols         %MultiLengths.datatype;  #IMPLIED
>
<![%xhtml-events.module;[
<!ATTLIST %frameset.qname;
      onload       %Script.datatype;        #IMPLIED
      onunload     %Script.datatype;        #IMPLIED
>
]]>

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

<!ENTITY % frame.content  "EMPTY" >
<!ELEMENT %frame.qname;  %frame.content; >
<!ATTLIST %frame.qname;
      %Core.attrib;
      longdesc     %URI.datatype;           #IMPLIED
      src          %URI.datatype;           #IMPLIED
      frameborder  ( 1 | 0 )                ’1’
      marginwidth  %Pixels.datatype;        #IMPLIED
      marginheight %Pixels.datatype;        #IMPLIED
      noresize     ( noresize )             #IMPLIED
      scrolling    ( yes | no | auto )      ’auto’
>

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

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

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

<!ENTITY % noframes.content "( %body.qname; )">
<!ELEMENT %noframes.qname;  %noframes.content; >
<!ATTLIST %noframes.qname;
      %Common.attrib;
>

<!-- end of xhtml-frames-1.mod -->
```

F.3.11. Target

```
<!-- ...................................................................... -->
<!-- XHTML Target Module  .................................................. -->
<!-- file: xhtml-target-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-target-1.mod,v 1.2 2000/09/27 22:28:18 radams Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Target 1.0//EN"
       SYSTEM "xhtml-target-1.mod"

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

<!-- Target

    target

     This module declares the 'target' attribute used for opening windows
-->

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

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

<!-- add 'target' attribute to 'area' element -->
<!ATTLIST %area.qname;
      target       %FrameTarget.datatype;   #IMPLIED
>

<!-- add 'target' attribute to 'link' element -->
```

```
<!ATTLIST %link.qname;
      target       %FrameTarget.datatype;   #IMPLIED
>

<!-- add 'target' attribute to 'form' element -->
<!ATTLIST %form.qname;
      target       %FrameTarget.datatype;   #IMPLIED
>

<!-- add 'target' attribute to 'base' element -->
<!ATTLIST %base.qname;
      target       %FrameTarget.datatype;   #IMPLIED
>

<!-- end of xhtml-target-1.mod -->
```

F.3.12. Iframe

```
<!-- ...................................................................... -->
<!-- XHTML IFrame Module  ................................................. -->
<!-- file: xhtml-iframe-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-iframe-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Inline Frame Element 1.0//EN"
       SYSTEM "xhtml-iframe-1.mod"

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

<!-- Inline Frames

        iframe

     This module declares the iframe element type and its attributes,
     used to create an inline frame within a document.
-->

<!-- Inline Frames .................................... -->

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

<!-- end of xhtml-iframe-1.mod -->
```

F.3.13. Intrinsic Events

```
<!-- ...................................................................... -->
<!-- XHTML Intrinsic Events Module  ....................................... -->
<!-- file: xhtml-events-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-events-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ENTITIES XHTML Intrinsic Events 1.0//EN"
       SYSTEM "xhtml-events-1.mod"

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

<!-- Intrinsic Event Attributes

     These are the event attributes defined in HTML 4.0,
     Section 18.2.3 "Intrinsic Events". This module must be
     instantiated prior to the Attributes Module but after
     the Datatype Module in the Modular Framework module.

    "Note: Authors of HTML documents are advised that changes
     are likely to occur in the realm of intrinsic events
     (e.g., how scripts are bound to events). Research in
     this realm is carried on by members of the W3C Document
     Object Model Working Group (see the W3C Web site at
     http://www.w3.org/ for more information)."
-->
<!-- NOTE: Because the ATTLIST declarations in this module occur
     before their respective ELEMENT declarations in other
     modules, there may be a dependency on this module that
     should be considered if any of the parameter entities used
     for element type names (eg., %foo.qname;) are redeclared.
-->

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

<!-- additional attributes on anchor element
-->
<!ATTLIST %a.qname;
     onfocus      %Script.datatype;         #IMPLIED
     onblur       %Script.datatype;         #IMPLIED
>

<!-- additional attributes on form element
-->
<!ATTLIST %form.qname;
      onsubmit     %Script.datatype;        #IMPLIED
      onreset      %Script.datatype;        #IMPLIED
>

<!-- additional attributes on label element
-->
<!ATTLIST %label.qname;
      onfocus      %Script.datatype;        #IMPLIED
      onblur       %Script.datatype;        #IMPLIED
>

<!-- additional attributes on input element
-->
<!ATTLIST %input.qname;
      onfocus      %Script.datatype;        #IMPLIED
      onblur       %Script.datatype;        #IMPLIED
      onselect     %Script.datatype;        #IMPLIED
      onchange     %Script.datatype;        #IMPLIED
>

<!-- additional attributes on select element
-->
<!ATTLIST %select.qname;
      onfocus      %Script.datatype;        #IMPLIED
      onblur       %Script.datatype;        #IMPLIED
      onchange     %Script.datatype;        #IMPLIED
>

<!-- additional attributes on textarea element
-->
<!ATTLIST %textarea.qname;
      onfocus      %Script.datatype;        #IMPLIED
      onblur       %Script.datatype;        #IMPLIED
      onselect     %Script.datatype;        #IMPLIED
      onchange     %Script.datatype;        #IMPLIED
>

<!-- additional attributes on button element
-->
<!ATTLIST %button.qname;
      onfocus      %Script.datatype;        #IMPLIED
      onblur       %Script.datatype;        #IMPLIED
>
```

```
<!-- additional attributes on body element
-->
<!ATTLIST %body.qname;
      onload       %Script.datatype;        #IMPLIED
      onunload     %Script.datatype;        #IMPLIED
>

<!-- additional attributes on area element
-->
<!ATTLIST %area.qname;
      onfocus      %Script.datatype;        #IMPLIED
      onblur       %Script.datatype;        #IMPLIED
>

<!-- end of xhtml-events-1.mod -->
```

F.3.14. Metainformation

```
<!-- ...................................................................... -->
<!-- XHTML Document Metainformation Module  .............................. -->
<!-- file: xhtml-meta-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-meta-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Metainformation 1.0//EN"
       SYSTEM "xhtml-meta-1.mod"

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

<!-- Meta Information

        meta

     This module declares the meta element type and its attributes,
     used to provide declarative document metainformation.
-->

<!-- meta: Generic Metainformation ..................... -->

<!ENTITY % meta.element  "INCLUDE" >
<![%meta.element;[
<!ENTITY % meta.content  "EMPTY" >
<!ENTITY % meta.qname  "meta" >
<!ELEMENT %meta.qname;  %meta.content; >
<!-- end of meta.element -->]]>

<!ENTITY % meta.attlist  "INCLUDE" >
<![%meta.attlist;[
<!ATTLIST %meta.qname;
      %XHTML.xmlns.attrib;
      %I18n.attrib;
      http-equiv   NMTOKEN                  #IMPLIED
      name         NMTOKEN                  #IMPLIED
      content      CDATA                    #REQUIRED
      scheme       CDATA                    #IMPLIED
>
<!-- end of meta.attlist -->]]>

<!-- end of xhtml-meta-1.mod -->
```

F.3.15. Scripting

```
<!-- ...................................................................... -->
<!-- XHTML Document Scripting Module  ..................................... -->
<!-- file: xhtml-script-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-script-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Scripting 1.0//EN"
       SYSTEM "xhtml-script-1.mod"

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

<!-- Scripting

        script, noscript

     This module declares element types and attributes used to provide
     support for executable scripts as well as an alternate content
     container where scripts are not supported.
-->

<!-- script: Scripting Statement ....................... -->

<!ENTITY % script.element  "INCLUDE" >
<![%script.element;[
<!ENTITY % script.content  "( #PCDATA )" >
<!ENTITY % script.qname  "script" >
<!ELEMENT %script.qname;  %script.content; >
<!-- end of script.element -->]]>

<!ENTITY % script.attlist  "INCLUDE" >
<![%script.attlist;[
<!ATTLIST %script.qname;
      %XHTML.xmlns.attrib;
      charset      %Charset.datatype;       #IMPLIED
      type         %ContentType.datatype;   #REQUIRED
      src          %URI.datatype;           #IMPLIED
      defer        ( defer )                #IMPLIED
      xml:space    ( preserve )             #FIXED 'preserve'
>
<!-- end of script.attlist -->]]>

<!-- noscript: No-Script Alternate Content ............. -->

<!ENTITY % noscript.element  "INCLUDE" >
<![%noscript.element;[
<!ENTITY % noscript.content
     "( %Block.mix; )+"
>
<!ENTITY % noscript.qname  "noscript" >
<!ELEMENT %noscript.qname;  %noscript.content; >
<!-- end of noscript.element -->]]>

<!ENTITY % noscript.attlist  "INCLUDE" >
<![%noscript.attlist;[
<!ATTLIST %noscript.qname;
      %Common.attrib;
>
<!-- end of noscript.attlist -->]]>

<!-- end of xhtml-script-1.mod -->
```

F.3.16. Stylesheet

```
<!-- ...................................................................... -->
<!-- XHTML Document Stylesheet Module  .................................... -->
<!-- file: xhtml-style-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-style-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//DTD XHTML Stylesheets 1.0//EN"
       SYSTEM "xhtml-style-1.mod"

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

<!-- Stylesheets

        style

     This module declares the style element type and its attributes,
     used to embed stylesheet information in the document head element.
-->

<!-- style: Stylesheet Information .................... -->

<!ENTITY % style.element  "INCLUDE" >
<![%style.element;[
<!ENTITY % style.content  "( #PCDATA )" >
<!ENTITY % style.qname  "style" >
<!ELEMENT %style.qname;  %style.content; >
<!-- end of style.element -->]]>

<!ENTITY % style.attlist  "INCLUDE" >
<![%style.attlist;[
<!ATTLIST %style.qname;
      %XHTML.xmlns.attrib;
      %title.attrib;
      %I18n.attrib;
      type         %ContentType.datatype;   #REQUIRED
      media        %MediaDesc.datatype;     #IMPLIED
      xml:space    ( preserve )             #FIXED 'preserve'
>
<!-- end of style.attlist -->]]>

<!-- end of xhtml-style-1.mod -->
```

F.3.17. Style Attribute

```
<!-- ...................................................................... -->
<!-- XHTML Inline Style Module  ........................................... -->
<!-- file: xhtml-inlstyle-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-inlstyle-1.mod,v 1.5 2000/09/20 14:57:39 ahby Exp $

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

       PUBLIC "-//W3C//ENTITIES XHTML Inline Style 1.0//EN"
       SYSTEM "xhtml-inlstyle-1.mod"

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

<!-- Inline Style

     This module declares the 'style' attribute, used to support inline
     style markup. This module must be instantiated prior to the XHTML
     Common Attributes module in order to be included in %Core.attrib;.
-->

<!ENTITY % style.attrib
     "style        CDATA                    #IMPLIED"
>


<!ENTITY % Core.extra.attrib
     "%style.attrib;"
>

<!-- end of xhtml-inlstyle-1.mod -->
```

F.3.18. Link

```
<!-- ...................................................................... -->
<!-- XHTML Link Element Module  ........................................... -->
<!-- file: xhtml-link-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-link-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Link Element 1.0//EN"
       SYSTEM "xhtml-link-1.mod"

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

<!-- Link element

        link

     This module declares the link element type and its attributes,
     which could (in principle) be used to define document-level links
     to external resources such as:

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

<!-- link: Media-Independent Link ...................... -->

<!ENTITY % link.element  "INCLUDE" >
<![%link.element;[
<!ENTITY % link.content  "EMPTY" >
<!ENTITY % link.qname  "link" >
<!ELEMENT %link.qname;  %link.content; >
<!-- end of link.element -->]]>

<!ENTITY % link.attlist  "INCLUDE" >
<![%link.attlist;[
<!ATTLIST %link.qname;
      %Common.attrib;
      charset      %Charset.datatype;       #IMPLIED
      href         %URI.datatype;           #IMPLIED
      hreflang     %LanguageCode.datatype;  #IMPLIED
      type         %ContentType.datatype;   #IMPLIED
      rel          %LinkTypes.datatype;     #IMPLIED
      rev          %LinkTypes.datatype;     #IMPLIED
      media        %MediaDesc.datatype;     #IMPLIED
>
<!-- end of link.attlist -->]]>

<!-- end of xhtml-link-1.mod -->
```

F.3.19. Base

```
<!-- ...................................................................... -->
<!-- XHTML Base Element Module  ........................................... -->
<!-- file: xhtml-base-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-base-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Base Element 1.0//EN"
       SYSTEM "xhtml-base-1.mod"

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

<!-- Base element

        base

     This module declares the base element type and its attributes,
     used to define a base URI against which relative URIs in the
     document will be resolved.

     Note that this module also redeclares the content model for
     the head element to include the base element.
-->

<!-- base: Document Base URI ........................... -->

<!ENTITY % base.element  "INCLUDE" >
<![%base.element;[
<!ENTITY % base.content  "EMPTY" >
<!ENTITY % base.qname  "base" >
<!ELEMENT %base.qname;  %base.content; >
<!-- end of base.element -->]]>

<!ENTITY % base.attlist  "INCLUDE" >
<![%base.attlist;[
<!ATTLIST %base.qname;
      %XHTML.xmlns.attrib;
      href         %URI.datatype;           #REQUIRED
>
<!-- end of base.attlist -->]]>

<!ENTITY % head.content
    "( %HeadOpts.mix;,
     ( ( %title.qname;, %HeadOpts.mix;, ( %base.qname;, %HeadOpts.mix; )? )
     | ( %base.qname;, %HeadOpts.mix;, ( %title.qname;, %HeadOpts.mix; ))))"
>

<!-- end of xhtml-base-1.mod -->
```

F.3.20. Name Identification

```
<!-- ...................................................................... -->
<!-- XHTML Name Identifier Module  ........................................ -->
<!-- file: xhtml-nameident-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-nameident-1.mod,v 1.5 2000/09/20 14:57:39 ahby Exp $

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

       PUBLIC "-//W3C//ELEMENTS XHTML Name Identifier 1.0//EN"
       SYSTEM "xhtml-nameident-1.mod"

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

<!-- Name Identifier

       'name' attribute on form, img, a, map, applet, frame, iframe

     This module declares the 'name' attribute on element types when
     it is used as a node identifier for legacy linking and scripting
     support. This does not include those instances when 'name' is used
     as a container for form control, property or metainformation names.

     This module should be instantiated following all modules it modifies.
-->

<!ENTITY % form.attlist  "IGNORE" >
<![%form.attlist;[
<!ATTLIST %form.qname;
      name         CDATA                    #IMPLIED
>
<!-- end of form.attlist -->]]>

<!ENTITY % img.attlist  "IGNORE" >
<![%img.attlist;[
<!ATTLIST %img.qname;
      name         CDATA                    #IMPLIED
>
<!-- end of img.attlist -->]]>

<!ENTITY % a.attlist  "IGNORE" >
<![%a.attlist;[
<!ATTLIST %a.qname;
      name         CDATA                    #IMPLIED
>
<!-- end of a.attlist -->]]>

<!ENTITY % map.attlist  "IGNORE" >
<![%map.attlist;[
<!ATTLIST %map.qname;
      name         CDATA                    #IMPLIED
>
<!-- end of map.attlist -->]]>

<!ENTITY % applet.attlist  "IGNORE" >
<![%applet.attlist;[
<!ATTLIST %applet.qname;
      name         CDATA                    #IMPLIED
>
<!-- end of applet.attlist -->]]>

<!ENTITY % frame.attlist  "IGNORE" >
<![%frame.attlist;[
<!ATTLIST %frame.qname;
      name         CDATA                    #IMPLIED
>
<!-- end of frame.attlist -->]]>

<!ENTITY % iframe.attlist  "IGNORE" >
<![%iframe.attlist;[
<!ATTLIST %iframe.qname;
      name         CDATA                    #IMPLIED
>
<!-- end of iframe.attlist -->]]>

<!-- end of xhtml-nameident.mod -->
```

F.3.21. Legacy

```
<!-- ...................................................................... -->
<!-- XHTML Legacy Markup Module ........................................... -->
<!-- file: xhtml-legacy-1.mod

     This is an extension of XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-legacy-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Legacy Markup 1.0//EN"
       SYSTEM "xhtml-legacy-1.mod"

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

<!-- HTML Legacy Markup

        font, basefont, center, s, strike, u, dir, menu, isindex

          (plus additional datatypes and attributes)

     This optional module declares additional markup for simple
     presentation-related markup based on features found in the
     HTML 4.0 Transitional and Frameset DTDs. This relies on
     inclusion of the Legacy Redeclarations module. This module
     also declares the frames, inline frames and object modules.

     This is to allow XHTML 1.1 documents to be transformed for
     display on HTML browsers where CSS support is inconsistent
     or unavailable.
-->
<!-- Constructing a Legacy DTD

     To construct a DTD driver obtaining a close approximation of
     the HTML 4.0 Transitional and Frameset DTDs, declare the Legacy
     Redeclarations module as the pre-framework redeclaration parameter
     entity (%xhtml-prefw-redecl.mod;) and INCLUDE its conditional section:

        ...
        <!ENTITY % xhtml-prefw-redecl.module "INCLUDE" >
        <![%xhtml-prefw-redecl.module;[
        <!ENTITY % xhtml-prefw-redecl.mod
            PUBLIC "-//W3C//ELEMENTS XHTML Legacy Redeclarations 1.0//EN"
                   "xhtml-legacy-redecl-1.mod" >
        %xhtml-prefw-redecl.mod;]]>

     Such a DTD should be named with a variant FPI and redeclare
     the value of the %XHTML.version; parameter entity to that FPI:

         "-//Your Name Here//DTD XHTML Legacy 1.1//EN"

     IMPORTANT:  see also the notes included in the Legacy Redeclarations
     Module for information on how to construct a DTD using this module.
-->


<!-- Additional Element Types .................................... -->

<!ENTITY % font.element  "INCLUDE" >
<![%font.element;[
<!ENTITY % font.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % font.qname  "font" >
<!ELEMENT %font.qname;  %font.content; >
<!-- end of font.element -->]]>

<!ENTITY % font.attlist  "INCLUDE" >
<![%font.attlist;[
<!ATTLIST %font.qname;
      %Core.attrib;
      %I18n.attrib;
      size         CDATA                    #IMPLIED
      color        %Color.datatype;         #IMPLIED
      face         CDATA                    #IMPLIED
>
<!-- end of font.attlist -->]]>

<!ENTITY % basefont.element  "INCLUDE" >
<![%basefont.element;[
<!ENTITY % basefont.content "EMPTY" >
<!ENTITY % basefont.qname  "basefont" >
<!ELEMENT %basefont.qname;  %basefont.content; >
<!-- end of basefont.element -->]]>

<!ENTITY % basefont.attlist  "INCLUDE" >
<![%basefont.attlist;[
<!ATTLIST %basefont.qname;
      %id.attrib;
      size         CDATA                    #REQUIRED
      color        %Color.datatype;         #IMPLIED
      face         CDATA                    #IMPLIED
>
<!-- end of basefont.attlist -->]]>

<!ENTITY % center.element  "INCLUDE" >
<![%center.element;[
<!ENTITY % center.content
     "( #PCDATA | %Flow.mix; )*"
>
<!ENTITY % center.qname  "center" >
<!ELEMENT %center.qname;  %center.content; >
<!-- end of center.element -->]]>

<!ENTITY % center.attlist  "INCLUDE" >
<![%center.attlist;[
<!ATTLIST %center.qname;
      %Common.attrib;
>
<!-- end of center.attlist -->]]>

<!ENTITY % s.element  "INCLUDE" >
<![%s.element;[
<!ENTITY % s.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % s.qname  "s" >
<!ELEMENT %s.qname;  %s.content; >
<!-- end of s.element -->]]>

<!ENTITY % s.attlist  "INCLUDE" >
<![%s.attlist;[
<!ATTLIST %s.qname;
      %Common.attrib;
>
<!-- end of s.attlist -->]]>

<!ENTITY % strike.element  "INCLUDE" >
<![%strike.element;[
<!ENTITY % strike.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % strike.qname  "strike" >
<!ELEMENT %strike.qname;  %strike.content; >
<!-- end of strike.element -->]]>

<!ENTITY % strike.attlist  "INCLUDE" >
<![%strike.attlist;[
<!ATTLIST %strike.qname;
      %Common.attrib;
>
<!-- end of strike.attlist -->]]>

<!ENTITY % u.element  "INCLUDE" >
```

```
<![%u.element;[
<!ENTITY % u.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % u.qname  "u" >
<!ELEMENT %u.qname;  %u.content; >
<!-- end of u.element -->]]>

<!ENTITY % u.attlist  "INCLUDE" >
<![%u.attlist;[
<!ATTLIST %u.qname;
      %Common.attrib;
>
<!-- end of u.attlist -->]]>

<!-- NOTE: the content model for <dir> in HTML 4 excluded %Block.mix;
-->
<!ENTITY % dir.element  "INCLUDE" >
<![%dir.element;[
<!ENTITY % dir.content
     "( %li.qname; )+"
>
<!ENTITY % dir.qname  "dir" >
<!ELEMENT %dir.qname;  %dir.content; >
<!-- end of dir.element -->]]>

<!ENTITY % dir.attlist  "INCLUDE" >
<![%dir.attlist;[
<!ATTLIST %dir.qname;
      %Common.attrib;
      compact      ( compact )              #IMPLIED
>
<!-- end of dir.attlist -->]]>

<!-- NOTE: the content model for <menu> in HTML 4 excluded %Block.mix;
-->
<!ENTITY % menu.element  "INCLUDE" >
<![%menu.element;[
<!ENTITY % menu.content
     "( %li.qname; )+"
>
<!ENTITY % menu.qname  "menu" >
<!ELEMENT %menu.qname;  %menu.content; >
<!-- end of menu.element -->]]>

<!ENTITY % menu.attlist  "INCLUDE" >
<![%menu.attlist;[
<!ATTLIST %menu.qname;
      %Common.attrib;
      compact      ( compact )              #IMPLIED
>
<!-- end of menu.attlist -->]]>

<!ENTITY % isindex.element  "INCLUDE" >
<![%isindex.element;[
<!ENTITY % isindex.content "EMPTY" >
<!ENTITY % isindex.qname  "isindex" >
<!ELEMENT %isindex.qname;  %isindex.content; >
<!-- end of isindex.element -->]]>

<!ENTITY % isindex.attlist  "INCLUDE" >
```

```
<![%isindex.attlist;[
<!ATTLIST %isindex.qname;
      %Core.attrib;
      %I18n.attrib;
      prompt       %Text.datatype;          #IMPLIED
>
<!-- end of isindex.attlist -->]]>


<!-- Additional Attributes ....................................... -->

<!-- Alignment attribute for Transitional use in HTML browsers
     (this functionality is generally well-supported in CSS,
     except within some contexts)
-->
<!ENTITY % align.attrib
     "align        ( left | center | right | justify ) #IMPLIED"
>

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

<!-- add ’target’ attribute to ’a’ element -->
<!ATTLIST %a.qname;
      target       %FrameTarget.datatype;   #IMPLIED
>

<!ATTLIST %applet.qname;
      align       ( top | middle | bottom | left | right ) #IMPLIED
      hspace      %Pixels.datatype;         #IMPLIED
      vspace      %Pixels.datatype;         #IMPLIED
>

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

<!ATTLIST %br.qname;
      clear        ( left | all | right | none ) ’none’
>

<!ATTLIST %caption.qname;
      %align.attrib;
>

<!ATTLIST %div.qname;
      %align.attrib;
>

<!ATTLIST %h1.qname;
      %align.attrib;
>

<!ATTLIST %h2.qname;
      %align.attrib;
>
```

```
<!ATTLIST %h3.qname;
      %align.attrib;
>

<!ATTLIST %h4.qname;
      %align.attrib;
>

<!ATTLIST %h5.qname;
      %align.attrib;
>

<!ATTLIST %h6.qname;
      %align.attrib;
>

<!ATTLIST %hr.qname;
      align        ( left | center | right ) #IMPLIED
      noshade      ( noshade )              #IMPLIED
      size         %Pixels.datatype;        #IMPLIED
      width        %Length.datatype;        #IMPLIED
>

<!ATTLIST %img.qname;
      align       ( top | middle | bottom | left | right ) #IMPLIED
      border      %Pixels.datatype;         #IMPLIED
      hspace      %Pixels.datatype;         #IMPLIED
      vspace      %Pixels.datatype;         #IMPLIED
>

<!ATTLIST %input.qname;
      %align.attrib;
>

<!ATTLIST %legend.qname;
      align        ( top | bottom | left | right ) #IMPLIED
>

<!ATTLIST %li.qname;
      type         CDATA                     #IMPLIED
      value        CDATA                     #IMPLIED
>

<!ATTLIST %object.qname;
      align        ( top | middle | bottom | left | right ) #IMPLIED
      border       %Pixels.datatype;         #IMPLIED
      hspace       %Pixels.datatype;         #IMPLIED
      vspace       %Pixels.datatype;         #IMPLIED
>

<!ATTLIST %ol.qname;
      type         CDATA                    #IMPLIED
      compact      ( compact )              #IMPLIED
      start        CDATA                    #IMPLIED
>

<!ATTLIST %p.qname;
      %align.attrib;
>
```

```
<!ATTLIST %pre.qname;
      width        %Length.datatype;        #IMPLIED
>

<!ATTLIST %script.qname;
      language     %ContentType.datatype;   #IMPLIED
>

<!ATTLIST %table.qname;
      %align.attrib;
      bgcolor      %Color.datatype;         #IMPLIED
>

<!ATTLIST %tr.qname;
      bgcolor     %Color.datatype;          #IMPLIED
>

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

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

<!ATTLIST %ul.qname;
      type         CDATA                    #IMPLIED
      compact      ( compact )              #IMPLIED
>

<!-- Frames Module ................................................ -->
<!ENTITY % xhtml-frames.module "IGNORE" >
<![%xhtml-frames.module;[
<!ENTITY % xhtml-frames.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Frames 1.0//EN"
            "xhtml-frames-1.mod" >
%xhtml-frames.mod;]]>

<!-- Inline Frames Module ......................................... -->
<!ENTITY % xhtml-iframe.module "INCLUDE" >
<![%xhtml-iframe.module;[
<!ENTITY % xhtml-iframe.mod
     PUBLIC "-//W3C//ELEMENTS XHTML Inline Frame Element 1.0//EN"
            "xhtml-iframe-1.mod" >
%xhtml-iframe.mod;]]>

<!-- end of xhtml-legacy-1.mod -->
```

F.4. 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.

F.4.1. Block Phrasal

```
<!-- ...................................................................... -->
<!-- XHTML Block Phrasal Module  .......................................... -->
<!-- file: xhtml-blkphras-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-blkphras-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Block Phrasal 1.0//EN"
       SYSTEM "xhtml-blkphras-1.mod"

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

<!-- Block Phrasal

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

     This module declares the elements and their attributes used to
     support block-level phrasal markup.
-->

<!ENTITY % address.element  "INCLUDE" >
<![%address.element;[
<!ENTITY % address.content
     "( #PCDATA | %Inline.mix; )*" >
<!ENTITY % address.qname  "address" >
<!ELEMENT %address.qname;  %address.content; >
<!-- end of address.element -->]]>

<!ENTITY % address.attlist  "INCLUDE" >
<![%address.attlist;[
<!ATTLIST %address.qname;
      %Common.attrib;
>
<!-- end of address.attlist -->]]>

<!ENTITY % blockquote.element  "INCLUDE" >
<![%blockquote.element;[
<!ENTITY % blockquote.content
     "( %Block.mix; )+"
>
<!ENTITY % blockquote.qname  "blockquote" >
<!ELEMENT %blockquote.qname;  %blockquote.content; >
<!-- end of blockquote.element -->]]>

<!ENTITY % blockquote.attlist  "INCLUDE" >
<![%blockquote.attlist;[
<!ATTLIST %blockquote.qname;
      %Common.attrib;
      cite         %URI.datatype;           #IMPLIED
>
<!-- end of blockquote.attlist -->]]>

<!ENTITY % pre.element  "INCLUDE" >
<![%pre.element;[
<!ENTITY % pre.content
     "( #PCDATA
      | %InlStruct.class;
      %InlPhras.class;
      | %tt.qname; | %i.qname; | %b.qname;
      %I18n.class;
      %Anchor.class;
      | %script.qname; | %map.qname;
      %Inline.extra; )*"
>
<!ENTITY % pre.qname  "pre" >
<!ELEMENT %pre.qname;  %pre.content; >
<!-- end of pre.element -->]]>

<!ENTITY % pre.attlist  "INCLUDE" >
<![%pre.attlist;[
<!ATTLIST %pre.qname;
      %Common.attrib;
      xml:space    ( preserve )             #FIXED ’preserve’
>
<!-- end of pre.attlist -->]]>

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

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

<!ENTITY % h1.element  "INCLUDE" >
<![%h1.element;[
<!ENTITY % h1.qname  "h1" >
<!ELEMENT %h1.qname;  %Heading.content; >
<!-- end of h1.element -->]]>

<!ENTITY % h1.attlist  "INCLUDE" >
<![%h1.attlist;[
<!ATTLIST %h1.qname;
      %Common.attrib;
>
<!-- end of h1.attlist -->]]>

<!ENTITY % h2.element  "INCLUDE" >
<![%h2.element;[
<!ENTITY % h2.qname  "h2" >
<!ELEMENT %h2.qname;  %Heading.content; >
<!-- end of h2.element -->]]>
```

```
<!ENTITY % h2.attlist  "INCLUDE" >
<![%h2.attlist;[
<!ATTLIST %h2.qname;
      %Common.attrib;
>
<!-- end of h2.attlist -->]]>

<!ENTITY % h3.element  "INCLUDE" >
<![%h3.element;[
<!ENTITY % h3.qname  "h3" >
<!ELEMENT %h3.qname;  %Heading.content; >
<!-- end of h3.element -->]]>

<!ENTITY % h3.attlist  "INCLUDE" >
<![%h3.attlist;[
<!ATTLIST %h3.qname;
      %Common.attrib;
>
<!-- end of h3.attlist -->]]>

<!ENTITY % h4.element  "INCLUDE" >
<![%h4.element;[
<!ENTITY % h4.qname  "h4" >
<!ELEMENT %h4.qname;  %Heading.content; >
<!-- end of h4.element -->]]>

<!ENTITY % h4.attlist  "INCLUDE" >
<![%h4.attlist;[
<!ATTLIST %h4.qname;
      %Common.attrib;
>
<!-- end of h4.attlist -->]]>

<!ENTITY % h5.element  "INCLUDE" >
<![%h5.element;[
<!ENTITY % h5.qname  "h5" >
<!ELEMENT %h5.qname;  %Heading.content; >
<!-- end of h5.element -->]]>

<!ENTITY % h5.attlist  "INCLUDE" >
<![%h5.attlist;[
<!ATTLIST %h5.qname;
      %Common.attrib;
>
<!-- end of h5.attlist -->]]>

<!ENTITY % h6.element  "INCLUDE" >
<![%h6.element;[
<!ENTITY % h6.qname  "h6" >
<!ELEMENT %h6.qname;  %Heading.content; >
<!-- end of h6.element -->]]>

<!ENTITY % h6.attlist  "INCLUDE" >
<![%h6.attlist;[
<!ATTLIST %h6.qname;
      %Common.attrib;
>
<!-- end of h6.attlist -->]]>

<!-- end of xhtml-blkphras-1.mod -->
```

F.4.2. Block Presentational

```
<!-- ...................................................................... -->
<!-- XHTML Block Presentation Module  ..................................... -->
<!-- file: xhtml-blkpres-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-blkpres-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Block Presentation 1.0//EN"
       SYSTEM "xhtml-blkpres-1.mod"

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

<!-- Block Presentational Elements

        hr

     This module declares the elements and their attributes used to
     support block-level presentational markup.
-->

<!ENTITY % hr.element  "INCLUDE" >
<![%hr.element;[
<!ENTITY % hr.content  "EMPTY" >
<!ENTITY % hr.qname  "hr" >
<!ELEMENT %hr.qname;  %hr.content; >
<!-- end of hr.element -->]]>

<!ENTITY % hr.attlist  "INCLUDE" >
<![%hr.attlist;[
<!ATTLIST %hr.qname;
      %Common.attrib;
>
<!-- end of hr.attlist -->]]>

<!-- end of xhtml-blkpres-1.mod -->
```

F.4.3. Block Structural

```
<!-- ...................................................................... -->
<!-- XHTML Block Structural Module  ....................................... -->
<!-- file: xhtml-blkstruct-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-blkstruct-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Block Structural 1.0//EN"
       SYSTEM "xhtml-blkstruct-1.mod"

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

<!-- Block Structural

        div, p

     This module declares the elements and their attributes used to
     support block-level structural markup.
-->

<!ENTITY % div.element  "INCLUDE" >
<![%div.element;[
<!ENTITY % div.content
     "( #PCDATA | %Flow.mix; )*"
>
<!ENTITY % div.qname  "div" >
<!ELEMENT %div.qname;  %div.content; >
<!-- end of div.element -->]]>

<!ENTITY % div.attlist  "INCLUDE" >
<![%div.attlist;[
<!-- end of div.attlist -->]]>
<!ATTLIST %div.qname;
      %Common.attrib;
>

<!ENTITY % p.element  "INCLUDE" >
<![%p.element;[
<!ENTITY % p.content
     "( #PCDATA | %Inline.mix; )*" >
<!ENTITY % p.qname  "p" >
<!ELEMENT %p.qname;  %p.content; >
<!-- end of p.element -->]]>

<!ENTITY % p.attlist  "INCLUDE" >
<![%p.attlist;[
<!ATTLIST %p.qname;
      %Common.attrib;
>
<!-- end of p.attlist -->]]>

<!-- end of xhtml-blkstruct-1.mod -->
```

F.4.4. Inline Phrasal

```
<!-- ...................................................................... -->
<!-- XHTML Inline Phrasal Module  ........................................ -->
<!-- file: xhtml-inlphras-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-inlphras-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Inline Phrasal 1.0//EN"
       SYSTEM "xhtml-inlphras-1.mod"

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

<!-- Inline Phrasal

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

     This module declares the elements and their attributes used to
     support inline-level phrasal markup.
-->

<!ENTITY % abbr.element  "INCLUDE" >
<![%abbr.element;[
<!ENTITY % abbr.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % abbr.qname  "abbr" >
<!ELEMENT %abbr.qname;  %abbr.content; >
<!-- end of abbr.element -->]]>

<!ENTITY % abbr.attlist  "INCLUDE" >
<![%abbr.attlist;[
<!ATTLIST %abbr.qname;
      %Common.attrib;
>
<!-- end of abbr.attlist -->]]>

<!ENTITY % acronym.element  "INCLUDE" >
<![%acronym.element;[
<!ENTITY % acronym.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % acronym.qname  "acronym" >
<!ELEMENT %acronym.qname;  %acronym.content; >
<!-- end of acronym.element -->]]>

<!ENTITY % acronym.attlist  "INCLUDE" >
<![%acronym.attlist;[
<!ATTLIST %acronym.qname;
      %Common.attrib;
```

```
>
<!-- end of acronym.attlist -->]]>

<!ENTITY % cite.element  "INCLUDE" >
<![%cite.element;[
<!ENTITY % cite.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % cite.qname  "cite" >
<!ELEMENT %cite.qname;  %cite.content; >
<!-- end of cite.element -->]]>

<!ENTITY % cite.attlist  "INCLUDE" >
<![%cite.attlist;[
<!ATTLIST %cite.qname;
      %Common.attrib;
>
<!-- end of cite.attlist -->]]>

<!ENTITY % code.element  "INCLUDE" >
<![%code.element;[
<!ENTITY % code.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % code.qname  "code" >
<!ELEMENT %code.qname;  %code.content; >
<!-- end of code.element -->]]>

<!ENTITY % code.attlist  "INCLUDE" >
<![%code.attlist;[
<!ATTLIST %code.qname;
      %Common.attrib;
>
<!-- end of code.attlist -->]]>

<!ENTITY % dfn.element  "INCLUDE" >
<![%dfn.element;[
<!ENTITY % dfn.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % dfn.qname  "dfn" >
<!ELEMENT %dfn.qname;  %dfn.content; >
<!-- end of dfn.element -->]]>

<!ENTITY % dfn.attlist  "INCLUDE" >
<![%dfn.attlist;[
<!ATTLIST %dfn.qname;
      %Common.attrib;
>
<!-- end of dfn.attlist -->]]>

<!ENTITY % em.element  "INCLUDE" >
<![%em.element;[
<!ENTITY % em.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % em.qname  "em" >
```

```
<!ELEMENT %em.qname;  %em.content; >
<!-- end of em.element -->]]>

<!ENTITY % em.attlist  "INCLUDE" >
<![%em.attlist;[
<!ATTLIST %em.qname;
      %Common.attrib;
>
<!-- end of em.attlist -->]]>

<!ENTITY % kbd.element  "INCLUDE" >
<![%kbd.element;[
<!ENTITY % kbd.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % kbd.qname  "kbd" >
<!ELEMENT %kbd.qname;  %kbd.content; >
<!-- end of kbd.element -->]]>

<!ENTITY % kbd.attlist  "INCLUDE" >
<![%kbd.attlist;[
<!ATTLIST %kbd.qname;
      %Common.attrib;
>
<!-- end of kbd.attlist -->]]>

<!ENTITY % q.element  "INCLUDE" >
<![%q.element;[
<!ENTITY % q.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % q.qname  "q" >
<!ELEMENT %q.qname;  %q.content; >
<!-- end of q.element -->]]>

<!ENTITY % q.attlist  "INCLUDE" >
<![%q.attlist;[
<!ATTLIST %q.qname;
      %Common.attrib;
      cite         %URI.datatype;           #IMPLIED
>
<!-- end of q.attlist -->]]>

<!ENTITY % samp.element  "INCLUDE" >
<![%samp.element;[
<!ENTITY % samp.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % samp.qname  "samp" >
<!ELEMENT %samp.qname;  %samp.content; >
<!-- end of samp.element -->]]>

<!ENTITY % samp.attlist  "INCLUDE" >
<![%samp.attlist;[
<!ATTLIST %samp.qname;
      %Common.attrib;
>
```

```
<!-- end of samp.attlist -->]]>

<!ENTITY % strong.element  "INCLUDE" >
<![%strong.element;[
<!ENTITY % strong.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % strong.qname  "strong" >
<!ELEMENT %strong.qname;  %strong.content; >
<!-- end of strong.element -->]]>

<!ENTITY % strong.attlist  "INCLUDE" >
<![%strong.attlist;[
<!ATTLIST %strong.qname;
      %Common.attrib;
>
<!-- end of strong.attlist -->]]>

<!ENTITY % var.element  "INCLUDE" >
<![%var.element;[
<!ENTITY % var.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % var.qname  "var" >
<!ELEMENT %var.qname;  %var.content; >
<!-- end of var.element -->]]>

<!ENTITY % var.attlist  "INCLUDE" >
<![%var.attlist;[
<!ATTLIST %var.qname;
      %Common.attrib;
>
<!-- end of var.attlist -->]]>

<!-- end of xhtml-inlphras-1.mod -->
```

F.4.5. Inline Presentational

```
<!-- ...................................................................... -->
<!-- XHTML Inline Presentation Module  .................................... -->
<!-- file: xhtml-inlpres-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-inlpres-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Inline Presentation 1.0//EN"
       SYSTEM "xhtml-inlpres-1.mod"

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

<!-- Inline Presentational Elements
```

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

     This module declares the elements and their attributes used to
     support inline-level presentational markup.
-->

<!ENTITY % b.element  "INCLUDE" >
<![%b.element;[
<!ENTITY % b.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % b.qname  "b" >
<!ELEMENT %b.qname;  %b.content; >
<!-- end of b.element -->]]>

<!ENTITY % b.attlist  "INCLUDE" >
<![%b.attlist;[
<!ATTLIST %b.qname;
      %Common.attrib;
>
<!-- end of b.attlist -->]]>

<!ENTITY % big.element  "INCLUDE" >
<![%big.element;[
<!ENTITY % big.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % big.qname  "big" >
<!ELEMENT %big.qname;  %big.content; >
<!-- end of big.element -->]]>

<!ENTITY % big.attlist  "INCLUDE" >
<![%big.attlist;[
<!ATTLIST %big.qname;
      %Common.attrib;
>
<!-- end of big.attlist -->]]>

<!ENTITY % i.element  "INCLUDE" >
<![%i.element;[
<!ENTITY % i.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % i.qname  "i" >
<!ELEMENT %i.qname;  %i.content; >
<!-- end of i.element -->]]>

<!ENTITY % i.attlist  "INCLUDE" >
<![%i.attlist;[
<!ATTLIST %i.qname;
      %Common.attrib;
>
<!-- end of i.attlist -->]]>

<!ENTITY % small.element  "INCLUDE" >
<![%small.element;[
<!ENTITY % small.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % small.qname  "small" >
<!ELEMENT %small.qname;  %small.content; >
<!-- end of small.element -->]]>

<!ENTITY % small.attlist  "INCLUDE" >
<![%small.attlist;[
<!ATTLIST %small.qname;
      %Common.attrib;
>
<!-- end of small.attlist -->]]>

<!ENTITY % sub.element  "INCLUDE" >
<![%sub.element;[
<!ENTITY % sub.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % sub.qname  "sub" >
<!ELEMENT %sub.qname;  %sub.content; >
<!-- end of sub.element -->]]>

<!ENTITY % sub.attlist  "INCLUDE" >
<![%sub.attlist;[
<!ATTLIST %sub.qname;
      %Common.attrib;
>
<!-- end of sub.attlist -->]]>

<!ENTITY % sup.element  "INCLUDE" >
<![%sup.element;[
<!ENTITY % sup.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % sup.qname  "sup" >
<!ELEMENT %sup.qname;  %sup.content; >
<!-- end of sup.element -->]]>

<!ENTITY % sup.attlist  "INCLUDE" >
<![%sup.attlist;[
<!ATTLIST %sup.qname;
      %Common.attrib;
>
<!-- end of sup.attlist -->]]>

<!ENTITY % tt.element  "INCLUDE" >
<![%tt.element;[
<!ENTITY % tt.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % tt.qname  "tt" >
<!ELEMENT %tt.qname;  %tt.content; >
<!-- end of tt.element -->]]>

<!ENTITY % tt.attlist  "INCLUDE" >
<![%tt.attlist;[
<!ATTLIST %tt.qname;
      %Common.attrib;
>
<!-- end of tt.attlist -->]]>

<!-- end of xhtml-inlpres-1.mod -->
```

F.4.6. Inline Structural

```
<!-- ...................................................................... -->
<!-- XHTML Inline Structural Module  ...................................... -->
<!-- file: xhtml-inlstruct-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-inlstruct-1.mod,v 1.9 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Inline Structural 1.0//EN"
       SYSTEM "xhtml-inlstruct-1.mod"

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

<!-- Inline Structural

        br, span

     This module declares the elements and their attributes
     used to support inline-level structural markup.
-->

<!-- br: forced line break ............................. -->

<!ENTITY % br.element  "INCLUDE" >
<![%br.element;[

<!ENTITY % br.content  "EMPTY" >
<!ENTITY % br.qname  "br" >
<!ELEMENT %br.qname;  %br.content; >

<!-- end of br.element -->]]>

<!ENTITY % br.attlist  "INCLUDE" >
<![%br.attlist;[
<!ATTLIST %br.qname;
      %Core.attrib;
>
<!-- end of br.attlist -->]]>

<!-- span: generic inline container .................... -->

<!ENTITY % span.element  "INCLUDE" >
<![%span.element;[
<!ENTITY % span.content
     "( #PCDATA | %Inline.mix; )*"
>
<!ENTITY % span.qname  "span" >
<!ELEMENT %span.qname;  %span.content; >
<!-- end of span.element -->]]>

<!ENTITY % span.attlist  "INCLUDE" >
<![%span.attlist;[
<!ATTLIST %span.qname;
      %Common.attrib;
>
<!-- end of span.attlist -->]]>

<!-- end of xhtml-inlstruct-1.mod -->
```

F.4.7. Param

```
<!-- ...................................................................... -->
<!-- XHTML Param Element Module  ..................................... -->
<!-- file: xhtml-param-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-param-1.mod,v 1.10 2000/09/20 14:57:39 ahby Exp $ SMI

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

       PUBLIC "-//W3C//ELEMENTS XHTML Param Element 1.0//EN"
       SYSTEM "xhtml-param-1.mod"

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

<!-- 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.
-->

<!-- param: Named Property Value ....................... -->

<!ENTITY % param.element  "INCLUDE" >
<![%param.element;[
<!ENTITY % param.content  "EMPTY" >
<!ENTITY % param.qname  "param" >
<!ELEMENT %param.qname;  %param.content; >
<!-- end of param.element -->]]>

<!ENTITY % param.attlist  "INCLUDE" >
<![%param.attlist;[
<!ATTLIST %param.qname;
```

```
      %XHTML.xmlns.attrib;
      %id.attrib;
      name         CDATA                    #REQUIRED
      value        CDATA                    #IMPLIED
      valuetype    ( data | ref | object )  'data'
      type         %ContentType.datatype;   #IMPLIED
>
<!-- end of param.attlist -->]]>

<!-- end of xhtml-param-1.mod -->
```

F.4.8. Qualified Names

```
<!-- ...................................................................... -->
<!-- XHTML Qname Module  ................................................... -->
<!-- file: xhtml-qname-1.mod

     This is XHTML, a reformulation of HTML as a modular XML application.
     Copyright 1998-2000 W3C (MIT, INRIA, Keio), All Rights Reserved.
     Revision: $Id: xhtml-qname-1.mod,v 1.13 2000/09/28 21:35:41 radams Exp $ SMI

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

       PUBLIC "-//W3C//ENTITIES XHTML Qualified Names 1.0//EN"
       SYSTEM "xhtml-qname-1.mod"

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

<!-- XHTML Qname (Qualified Name) Module

     This module is contained in two parts, labeled Section 'A' and 'B':

       Section A declares parameter entities to support namespace-
       qualified names, namespace declarations, and name prefixing
       for XHTML and extensions.

       Section B declares parameter entities used to provide
       namespace-qualified names for all XHTML element types:

         %applet.qname;   the xmlns-qualified name for <applet>
         %base.qname;     the xmlns-qualified name for <base>
         ...

     XHTML extensions would create a module similar to this one.
     Included in the XHTML distribution is a template module
     ('template-qname-1.mod') suitable for this purpose.
-->

<!-- Section A: XHTML XML Namespace Framework :::::::::::::::::::: -->

<!-- 1. Declare a %XHTML.prefixed; conditional section keyword, used
        to activate namespace prefixing. The default value should
        inherit '%NS.prefixed;' from the DTD driver, so that unless
        overridden, the default behaviour follows the overall DTD
        prefixing scheme.
-->
<!ENTITY % NS.prefixed "IGNORE" >
<!ENTITY % XHTML.prefixed "%NS.prefixed;" >

<!-- 2. Declare a parameter entity (eg., %XHTML.xmlns;) containing
        the URI reference used to identify the XHTML namespace:
-->
<!ENTITY % XHTML.xmlns  "http://www.w3.org/1999/xhtml" >

<!-- 3. Declare parameter entities (eg., %XHTML.prefix;) containing
        the default namespace prefix string(s) to use when prefixing
        is enabled. This may be overridden in the DTD driver or the
        internal subset of an document instance. If no default prefix
        is desired, this may be declared as an empty string.

     NOTE: As specified in [XMLNAMES], the namespace prefix serves
     as a proxy for the URI reference, and is not in itself significant.
-->
<!ENTITY % XHTML.prefix  "" >

<!-- 4. Declare parameter entities (eg., %XHTML.pfx;) containing the
        colonized prefix(es) (eg., '%XHTML.prefix;:') used when
        prefixing is active, an empty string when it is not.
-->
<![%XHTML.prefixed;[
<!ENTITY % XHTML.pfx  "%XHTML.prefix;:" >
]]>
<!ENTITY % XHTML.pfx  "" >

<!-- declare qualified name extensions here ............ -->
<!ENTITY % xhtml-qname-extra.mod "" >
%xhtml-qname-extra.mod;

<!-- 5. The parameter entity %XHTML.xmlns.extra.attrib; may be
        redeclared to contain any non-XHTML namespace declaration
        attributes for namespaces embedded in XHTML. The default
        is an empty string.  XLink should be included here if used
        in the DTD.
-->
<!ENTITY % XHTML.xmlns.extra.attrib "" >

<!-- The remainder of Section A is only followed in XHTML, not extensions. -->

<!-- Declare a parameter entity %NS.decl.attrib; containing
     all XML Namespace declarations used in the DTD, plus the
     xmlns declaration for XHTML, its form dependent on whether
     prefixing is active.
-->
<![%XHTML.prefixed;[
<!ENTITY % NS.decl.attrib
     "xmlns:%XHTML.prefix;  %URI.datatype;   #FIXED '%XHTML.xmlns;'
      %XHTML.xmlns.extra.attrib;"
>
]]>
<!ENTITY % NS.decl.attrib
     "%XHTML.xmlns.extra.attrib;"
```

```
>

<!-- This is a placeholder for future XLink support.
-->
<!ENTITY % XLINK.xmlns.attrib "" >

<!-- Declare a parameter entity %NS.decl.attrib; containing all
     XML namespace declaration attributes used by XHTML, including
     a default xmlns attribute when prefixing is inactive.
-->
<![%XHTML.prefixed;[
<!ENTITY % XHTML.xmlns.attrib
     "%NS.decl.attrib;
      %XLINK.xmlns.attrib;"
>
]]>
<!ENTITY % XHTML.xmlns.attrib
     "xmlns        %URI.datatype;           #FIXED '%XHTML.xmlns;'
      %XLINK.xmlns.attrib;"
>

<!-- placeholder for qualified name redeclarations -->
<!ENTITY % xhtml-qname.redecl "" >
%xhtml-qname.redecl;

<!-- Section B: XHTML Qualified Names ::::::::::::::::::::::::::::: -->

<!-- 6. This section declares parameter entities used to provide
        namespace-qualified names for all XHTML element types.
-->

<!-- module:  xhtml-applet-1.mod -->
<!ENTITY % applet.qname  "%XHTML.pfx;applet" >

<!-- module:  xhtml-base-1.mod -->
<!ENTITY % base.qname    "%XHTML.pfx;base" >

<!-- module:  xhtml-bdo-1.mod -->
<!ENTITY % bdo.qname     "%XHTML.pfx;bdo" >

<!-- module:  xhtml-blkphras-1.mod -->
<!ENTITY % address.qname "%XHTML.pfx;address" >
<!ENTITY % blockquote.qname  "%XHTML.pfx;blockquote" >
<!ENTITY % pre.qname     "%XHTML.pfx;pre" >
<!ENTITY % h1.qname      "%XHTML.pfx;h1" >
<!ENTITY % h2.qname      "%XHTML.pfx;h2" >
<!ENTITY % h3.qname      "%XHTML.pfx;h3" >
<!ENTITY % h4.qname      "%XHTML.pfx;h4" >
<!ENTITY % h5.qname      "%XHTML.pfx;h5" >
<!ENTITY % h6.qname      "%XHTML.pfx;h6" >

<!-- module:  xhtml-blkpres-1.mod -->
<!ENTITY % hr.qname      "%XHTML.pfx;hr" >

<!-- module:  xhtml-blkstruct-1.mod -->
<!ENTITY % div.qname     "%XHTML.pfx;div" >
<!ENTITY % p.qname       "%XHTML.pfx;p" >
```

```
<!-- module:  xhtml-edit-1.mod -->
<!ENTITY % ins.qname     "%XHTML.pfx;ins" >
<!ENTITY % del.qname     "%XHTML.pfx;del" >

<!-- module:  xhtml-form-1.mod -->
<!ENTITY % form.qname    "%XHTML.pfx;form" >
<!ENTITY % label.qname   "%XHTML.pfx;label" >
<!ENTITY % input.qname   "%XHTML.pfx;input" >
<!ENTITY % select.qname  "%XHTML.pfx;select" >
<!ENTITY % optgroup.qname  "%XHTML.pfx;optgroup" >
<!ENTITY % option.qname  "%XHTML.pfx;option" >
<!ENTITY % textarea.qname  "%XHTML.pfx;textarea" >
<!ENTITY % fieldset.qname  "%XHTML.pfx;fieldset" >
<!ENTITY % legend.qname  "%XHTML.pfx;legend" >
<!ENTITY % button.qname  "%XHTML.pfx;button" >

<!-- module:  xhtml-hypertext-1.mod -->
<!ENTITY % a.qname       "%XHTML.pfx;a" >

<!-- module:  xhtml-image-1.mod -->
<!ENTITY % img.qname     "%XHTML.pfx;img" >

<!-- module:  xhtml-inlphras-1.mod -->
<!ENTITY % abbr.qname    "%XHTML.pfx;abbr" >
<!ENTITY % acronym.qname "%XHTML.pfx;acronym" >
<!ENTITY % cite.qname    "%XHTML.pfx;cite" >
<!ENTITY % code.qname    "%XHTML.pfx;code" >
<!ENTITY % dfn.qname     "%XHTML.pfx;dfn" >
<!ENTITY % em.qname      "%XHTML.pfx;em" >
<!ENTITY % kbd.qname     "%XHTML.pfx;kbd" >
<!ENTITY % q.qname       "%XHTML.pfx;q" >
<!ENTITY % samp.qname    "%XHTML.pfx;samp" >
<!ENTITY % strong.qname  "%XHTML.pfx;strong" >
<!ENTITY % var.qname     "%XHTML.pfx;var" >

<!-- module:  xhtml-inlpres-1.mod -->
<!ENTITY % b.qname       "%XHTML.pfx;b" >
<!ENTITY % big.qname     "%XHTML.pfx;big" >
<!ENTITY % i.qname       "%XHTML.pfx;i" >
<!ENTITY % small.qname   "%XHTML.pfx;small" >
<!ENTITY % sub.qname     "%XHTML.pfx;sub" >
<!ENTITY % sup.qname     "%XHTML.pfx;sup" >
<!ENTITY % tt.qname      "%XHTML.pfx;tt" >

<!-- module:  xhtml-inlstruct-1.mod -->
<!ENTITY % br.qname      "%XHTML.pfx;br" >
<!ENTITY % span.qname    "%XHTML.pfx;span" >

<!-- module:  xhtml-ismap-1.mod (also csismap, ssismap) -->
<!ENTITY % map.qname     "%XHTML.pfx;map" >
<!ENTITY % area.qname    "%XHTML.pfx;area" >

<!-- module:  xhtml-link-1.mod -->
<!ENTITY % link.qname    "%XHTML.pfx;link" >

<!-- module:  xhtml-list-1.mod -->
```

```
<!ENTITY % dl.qname      "%XHTML.pfx;dl" >
<!ENTITY % dt.qname      "%XHTML.pfx;dt" >
<!ENTITY % dd.qname      "%XHTML.pfx;dd" >
<!ENTITY % ol.qname      "%XHTML.pfx;ol" >
<!ENTITY % ul.qname      "%XHTML.pfx;ul" >
<!ENTITY % li.qname      "%XHTML.pfx;li" >

<!-- module:  xhtml-meta-1.mod -->
<!ENTITY % meta.qname    "%XHTML.pfx;meta" >

<!-- module:  xhtml-param-1.mod -->
<!ENTITY % param.qname   "%XHTML.pfx;param" >

<!-- module:  xhtml-object-1.mod -->
<!ENTITY % object.qname  "%XHTML.pfx;object" >

<!-- module:  xhtml-script-1.mod -->
<!ENTITY % script.qname  "%XHTML.pfx;script" >
<!ENTITY % noscript.qname  "%XHTML.pfx;noscript" >

<!-- module:  xhtml-struct-1.mod -->
<!ENTITY % html.qname    "%XHTML.pfx;html" >
<!ENTITY % head.qname    "%XHTML.pfx;head" >
<!ENTITY % title.qname   "%XHTML.pfx;title" >
<!ENTITY % body.qname    "%XHTML.pfx;body" >

<!-- module:  xhtml-style-1.mod -->
<!ENTITY % style.qname   "%XHTML.pfx;style" >

<!-- module:  xhtml-table-1.mod -->
<!ENTITY % table.qname   "%XHTML.pfx;table" >
<!ENTITY % caption.qname "%XHTML.pfx;caption" >
<!ENTITY % thead.qname   "%XHTML.pfx;thead" >
<!ENTITY % tfoot.qname   "%XHTML.pfx;tfoot" >
<!ENTITY % tbody.qname   "%XHTML.pfx;tbody" >
<!ENTITY % colgroup.qname  "%XHTML.pfx;colgroup" >
<!ENTITY % col.qname     "%XHTML.pfx;col" >
<!ENTITY % tr.qname      "%XHTML.pfx;tr" >
<!ENTITY % th.qname      "%XHTML.pfx;th" >
<!ENTITY % td.qname      "%XHTML.pfx;td" >


<!-- Provisional XHTML 2.0 Qualified Names  ...................... -->

<!-- module:  xhtml-image-2.mod -->
<!ENTITY % alt.qname     "%XHTML.pfx;alt" >

<!-- end of xhtml-qname-1.mod -->
```

G. References

This appendix is *normative*.

G.1. Normative References

[CSS2]
"Cascading Style Sheets, level 2 (CSS2) Specification", B. Bos, H. W. Lie, C. Lilley, I. Jacobs, 12 May 1998.
Available at: http://www.w3.org/TR/1998/REC-CSS2-19980512

[DOM]
"Document Object Model (DOM) Level 1 Specification", Lauren Wood *et al.*, 1 October 1998.
Available at: http://www.w3.org/TR/REC-DOM-Level-1-19981001

[HTML4]
HTML 4.01 Specification: W3C Recommendation, Dave Raggett, Arnaud Le Hors, Ian Jacobs, 24 December 1999.
See: http://www.w3.org/TR/1999/REC-html401-19991224

[ISO10646]
"Information Technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane", ISO/IEC 10646-1:1993. This reference refers to a set of codepoints that may evolve as new characters are assigned to them. This reference therefore includes future amendments as long as they do **not** change character assignments up to and including the first five amendments to ISO/IEC 10646-1:1993. Also, this reference assumes that the character sets defined by ISO 10646 and Unicode remain character-by-character equivalent. This reference also includes future publications of other parts of 10646 (i.e., other than Part 1) that define characters in planes 1-16.

[MathML]
Mathematical Markup Language 1.01, Patrick Ion, et al. 7 July 1999.
See: http://www.w3.org/1999/07/REC-MathML-19990707

[RFC1766]
"Tags for the Identification of Languages", H. Alvestrand, March 1995. RFC1766 is expected to be updated by http://www.ietf.org/internet-drafts/draft-alvestrand-lang-tag-v2-04.txt, currently a work in progress.

[RFC1808]
Relative Uniform Resource Locators, R. Fielding.
See: http://www.ietf.org/rfc/rfc1808.txt

[RFC2045]
"Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed and N. Borenstein, November 1996. Note that this RFC obsoletes RFC1521, RFC1522, and RFC1590.

[SGML]
Information Processing -- Text and Office Systems -- Standard Generalized Markup Language (SGML), ISO 8879:1986.
Please consult http://www.iso.ch/cate/d16387.html for information about the standard, or

http://www.oasis-open.org/cover/general.html#overview about SGML.

[SMIL]
Synchronized Multimedia Integration Language (SMIL) 1.0 Specification, Philipp Hoschka, 15 June 1998.
See: http://www.w3.org/TR/1998/REC-smil-19980615

[SRGB]
"A Standard Default Color Space for the Internet", version 1.10, M. Stokes, M. Anderson, S. Chandrasekar, and R. Motta, 5 November 1996. This document is http://www.w3.org/Graphics/Color/sRGB

[URI]
Uniform Resource Identifiers (URI): Generic Syntax, T. Berners-Lee, R. Fielding, L. Masinter, August 1998.
See: http://www.ietf.org/rfc/rfc2396.txt. This RFC updates RFC 1738 [URL] [p.168] and [RFC1808] [p.167] .

[URL]
IETF RFC 1738, Uniform Resource Locators (URL), T. Berners-Lee, L. Masinter, M. McCahill.
See: http://www.ietf.org/rfc/rfc1738.txt

[XHTML1]
XHTML 1.0: The Extensible HyperText Markup Language, Steven Pemberton, et al., 26 January 2000.
See: http://www.w3.org/TR/2000/REC-xhtml1-20000126

[XML]
Extensible Markup Language (XML) 1.0: W3C Recommendation, Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, 10 February 1998.
See: http://www.w3.org/TR/1998/REC-xml-19980210

[XMLNAMES]
"Namespaces in XML", T. Bray, D. Hollander, A. Layman, 14 January 1999.
XML namespaces provide a simple method for qualifying names used in XML documents by associating them with namespaces identified by URI.
Available at: http://www.w3.org/TR/1999/REC-xml-names-19990114

[XMLSCHEMA]
XML Schema Part 1: Structures Henry S. Thompson, et al., 17 December 1999
See: http://www.w3.org/TR/1999/WD-xhtmlschema-1-19991217

H. Design Goals

This appendix is *informative*.

There are four major design goals for the modularization framework for XHTML:

- [G1] To group semantically related parts of XHTML together.
- [G2] Using DTD technology, to support the greation of related languages (subsets, supersets) for specific purposes (small devices, special-purpose devices), while guaranteeing commonality of the overlapping parts.
- [G3] To facilitate future development by allowing parts of the language to be replaced by improved modules (for instance, forms) without disturbing the rest of the language.
- [G4] To encourage and facilitate the reuse of modules in other languages.

H.1. Requirements

The design goals listed in the previous section lead to a large number of requirements for the modularization framework. These requirements, summarized in this section, can be further classified according to the major features of the framework to be described.

H.1.1. Granularity

Collectively the requirements in this section express the desire that the modules defined within the framework hit the right level of granularity:

- [R1.1] Abstract modules should promote and maintain content portability.
- [R1.2] Abstract modules should promote platform profile standardization.
- [R1.3] Abstract modules should be large enough to promote interoperability.
- [R1.4] Abstract modules should be small enough to avoid the need for subsets.
- [R1.5] Abstract modules should collect elements with similar or related semantics.
- [R1.6] Abstract modules should separate elements with dissimilar or unrelated semantics.
- [R1.7] Modules should be small enough to allow single element document type modules.

H.1.2. Compthird

The compactness requirements listed here are intended to ensure that the modularization framework be able to express the right set of target modules required by the communities that will be served by the framework:

- [R2.1] The module framework should allow construction of abstract modules for XHTML 1.0.
- [R2.2] The module framework should allow construction of abstract modules that closely approximate HTML 4.0.
- [R2.3] The module framework should allow construction of abstract modules for other W3C Recommendations.
- [R2.4] The module framework should allow construction of abstract modules for other XML

document types.
- [R2.5] The module framework should allow construction of abstract modules for a wide range of platform profiles.

H.1.3. Ease of Use

The modularization framework will only receive widespread adoption if it describes mechanisms that make it easy for our target audience to use the framework:

- [R3.1] The module framework should make it easy for document type designers to subset and extend XHTML abstract modules.
- [R3.2] The module framework should make it easy for document type designers to create abstract modules for other XML document types.
- [R3.3] The module framework should make it easy for document authors to validate elements from different abstract modules.

H.1.4. Compatibility

The intent of this document is that the modularization framework described here should work well with the XML and other standards being developed by the W3C Working Groups:

- [R4.1] The module framework should strictly conform to the XML 1.0 Recommendation.
- [R4.2] The module framework should be compatible with the XML linking specification.
- [R4.3] The module framework should be compatible with the XML stylesheet specification.
- [R4.4] The module framework should be able to adopt new W3C recommendations where appropriate.
- [R4.5] The module framework should not depend on W3C work in progress.
- [R4.6] The module framework should not depend on work done outside W3C.

H.1.5. Conformance

The effectiveness of the framework will also be measured by how easy it is to test the behavior of modules developed according to the framework, and to test the documents that employ those modules for validation:

- [R5.1] It should be possible to validate documents constructed using elements and attributes from abstract modules.
- [R5.2] It should be possible to explicitly describe the behavior of elements and attributes from abstract modules.
- [R5.3] It should be possible to verify the behavior of elements and attributes from abstract modules.
- [R5.4] It should be possible to verify a hybrid document type as an XHTML document type.
- [R5.5] Modules defined in accordance with the methods in this document shall not duplicate the names of elements or parameter entities defined in XHTML modules.