Abstract

The XHTML Basic document type includes the minimal set of modules required to be an XHTML host language document type, and in addition it includes images, forms, basic tables, and object support. It is designed for Web clients that do not support the full set of XHTML features; for example, Web clients such as mobile phones, PDAs, pagers, and set top boxes. The document type is rich enough for content authoring.
XHTML Basic is designed as a common base that may be extended. The goal of XHTML Basic is to serve as a common language supported by various kinds of user agents.

This revision, 1.1 Second Edition, supercedes version 1.1 as defined in [http://www.w3.org/TR/2008/REC-xhtml-basic-20080729]. In this revision, an XML Schema implementation and the lang attribute have been added. In the update from version 1.0 to version 1.1, several new features were incorporated into the language in order to better serve the small-device community that is this language’s major user:

1. XHTML Forms (defined in [XHTMLMOD][p.23])
2. Intrinsic Events (defined in [XHTMLMOD][p.23])
3. The value attribute for the li element (defined in [XHTMLMOD][p.23])
4. The target attribute (defined in [XHTMLMOD][p.23])
5. The style element (defined in [XHTMLMOD][p.23])
6. The style attribute (defined in [XHTMLMOD][p.23])
7. XHTML Presentation module (defined in [XHTMLMOD][p.23])
8. The inputmode attribute (defined in Section 5[p.15] of this document)

The document type definition is implemented using XHTML modules as defined in "XHTML Modularization" [XHTMLMOD][p.23].

Status of this Document

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

This document is a W3C Proposed Edited Recommendation. If approved, it will supersede the previous edition of XHTML Basic. The only changes in this Proposed Edited Recommendation are to add an XML Schema implementation of the markup language and integrate the lang attribute to increase compatibility with User Agents and Assistive Technologies. A version that shows the specific changes from the previous Recommendation is available in diff-marked form.

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

W3C Advisory Committee Members are invited to send formal review comments on this Proposed Edited Recommendation to the W3C Team until 11 November 2010. Members of the W3C Advisory Committee will find the appropriate review form for this document by consulting their list of current WBS questionnaires.

This document has been produced by the W3C XHTML2 Working Group as part of the W3C HTML Activity. Please see the Working Group’s implementation report.
Please send comments about this document to www-html-editor@w3.org (archive). It is inappropriate to send discussion email to this address. Public discussion may take place on www-html@w3.org (archive).

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

Table of Contents

- 3 -
1. Introduction

1.1. XHTML for Small Information Appliances

HTML 4 is a powerful language for authoring Web content, but its design does not take into consideration issues pertinent to small devices, including the implementation cost (in power, memory, etc.) of the full feature set. Consumer devices with limited resources cannot generally afford to implement the full feature set of HTML 4. Requiring a full-fledged computer for access to the World Wide Web excludes a large portion of the population from consumer device access of online information and services.

Because there are many ways to subset HTML, there are many almost identical subsets defined by organizations and companies. Without a common base set of features, developing applications for a wide range of Web clients is difficult.

The motivation for XHTML Basic is to provide an XHTML document type that can be shared across communities (e.g. desktop, TV, and mobile phones), and that is rich enough to be used for simple content authoring. New community-wide document types can be defined by extending XHTML Basic in such a way that XHTML Basic documents are in the set of valid documents of the new document type. Thus an XHTML Basic document can be presented on the maximum number of Web clients.

The document type definition for XHTML Basic is implemented based on the XHTML modules defined in XHTML Modularization [XHTMLMOD][p.23].

For information on best practices for mobile content, we refer you to [MOBILEBP][p.24].

1.2. Background and Requirements

Information appliances are targeted for particular uses. They support the features they need for the functions they are designed to fulfill. The following are examples of different information appliances:

- Mobile phones
- Televisions
- PDAs
- Vending machines
- Pagers
- Car navigation systems
- Mobile game machines
- Digital book readers
- Smart watches

Existing subsets and variants of HTML for these clients include Compact HTML [CHTML][p.23], the Wireless Markup Language [WML][p.24], and the “HTML 4.0 Guidelines for Mobile Access” [GUIDELINES][p.23]. The common features found in these document types include:
1.3. Design Rationale

This section explains why certain HTML features are not part of XHTML Basic.

1.3.1. Presentation

Many simple Web clients cannot display fonts other than monospace. Bi-directional text, bold faced font, and other text extension elements are not supported.

It is recommended that style sheets be used to create a presentation that is appropriate for the device.

1.3.2. Tables

Basic XHTML tables ([XHTMLMOD][p.23], section 5.6.1) are supported, but tables can be difficult to display on small devices. It is recommended that content developers follow the Web Content Accessibility Guidelines 1.0 for creating accessible tables ([WCAG10][p.24], Guideline 5). Note that in the Basic Tables Module, nesting of tables is prohibited.
1.3.3. Frames

Frames are not supported. Frames depend on a screen interface and may not be applicable to some small appliances like phones, pagers, and watches.
2. Conformance

This section is normative.

2.1. Document Conformance

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

1. The document must conform to the constraints expressed in Appendix B [p. 25] and Appendix C [p. 35].
2. The root element of the document must be `<html>`.
3. The name of the default namespace on the root element must be the XHTML namespace name, http://www.w3.org/1999/xhtml.

The start tag MAY also contain the declaration of the XML Schema Instance Namespace and an XML Schema Instance schemaLocation attribute [XMLSCHEMA [p. 23]]. Such an attribute would associate the XHTML namespace http://www.w3.org/1999/xhtml with the XML Schema at the URI http://www.w3.org/MarkUp/SCHEMA/xhtml-basic11.xsd.

4. There must be a DOCTYPE declaration in the document prior to the root element. If present, the public identifier included in the DOCTYPE declaration must reference the DTD found in Appendix B [p. 25] using its Formal Public Identifier. The system identifier may be modified appropriately.

```xml
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.1//EN"
"http://www.w3.org/TR/xhtml-basic/xhtml-basic11.dtd">
```

5. The DTD subset must not be used to override any parameter entities in the DTD.

XHTML Basic 1.1 documents SHOULD be labeled with the Internet Media Type "application/xhtml+xml" as defined in [RFC3236 [p. 23]]. For further information on using media types with XHTML, see the informative note [XHTMLEMIME [p. 24]].

2.2. User Agent Conformance

The user agent must conform to the 'User Agent Conformance' section of the XHTML 1.0 specification ([XHTML1 [p. 23], section 3.2]).
3. The XHTML Basic Document Type

This section is normative.

The XHTML Basic document type is defined as a set of XHTML modules. All XHTML modules are defined in the "XHTML Modularization" specification [XHTMLMOD][p.23].

XHTML Basic consists of the following XHTML modules:

- **Structure Module**
  - body, head, html, title

- **Text Module**
  - abbr, acronym, address, blockquote, br, cite, code, dfn, div, em, h1, h2, h3, h4, h5, h6, kbd, p, pre, q, samp, span, strong, var

- **Hypertext Module**
  - a

- **List Module**
  - dl, dt, dd, ol, ul, li

- **Forms Module**
  - button, fieldset, form, input, label, legend, select, optgroup, option, textarea

- **Basic Tables Module**
  - caption, table, td, th, tr

- **Image Module**
  - img

- **Object Module**
  - object, param

- **Presentation module**
  - b, big, hr, i, small, sub, sup, tt

- **Metainformation Module**
  - meta

- **Link Module**
  - link

- **Base Module**
  - base

- **Intrinsic Events module**
  - Events attributes

- **Scripting module**
  - script and noscript elements

- **Stylesheet module**
  - style element

- **Style Attribute Module** [Deprecated]
  - style attribute

- **Target Module**
  - target attribute.
Note:

1. The target attribute is designed to be a general hook for binding to an external environment (such as Frames, multiple windows, browser-tabbed windows); when there is no such external environment bound to the user agent, the user agent can ignore the target attribute. When there is an external environment bound, the conformance requirements for the target attribute are defined in each environment.

2. The content author needs to be aware that the user agent behavior for the target attribute depends on multiple factors such as the existence of an environment binding, restrictions of available resources, existence of other applications and user preferences (such as pop-up blockers), and implementation-dependent design decisions. When there is no external environmental conformance, it is recommended that authors do not depend on use of the target attribute.

3. It should be noted that any implementation-dependent use of the target attribute might impede interoperability.

This specification also adds the lang attribute to the I18N attribute collection as defined in [XHTMLMOD][p.23]. The lang attribute is defined in [HTML4][p.23]. When this attribute and the xml:lang attribute are specified on the same element, the xml:lang attribute takes precedence. When both lang and xml:lang are specified on the same element, they SHOULD have the same value.

(*) = This module is a required XHTML Host Language module.

XHTML Basic also uses the [XHTML inputmode Attribute Module][p.15], as defined in this specification. This module adds the inputmode attribute to the input and textarea elements of the XHTML Forms Module.

Finally, XHTML Basic adds the value attribute to the li element of the XHTML List Module.

An XML 1.0 DTD is available in [Appendix B.][p.25] An XML Schema implementation is available in [Appendix C.][p.35]
4. How to Use XHTML Basic

Although XHTML Basic can be used as it is - a simple XHTML language with text, links, and images - the intention of its simple design is for use as a host language. A host language can contain a mix of vocabularies all rolled into one document type. It is natural that XHTML is the host language, since that is what most Web developers are used to.

When markup from other languages is added to XHTML Basic, the resulting document type will be an extension of XHTML Basic. Content developers can develop for XHTML Basic or take advantage of the extensions. The goal of XHTML Basic is to serve as a common language supported by various kinds of user agents.
5. XHTML inputmode Attribute Module

This section is normative.

This section was originally a component of [XForms 1.0][24], and was written by Martin Duerst.

The inputmode Attribute Module defines the inputmode attribute.

**inputmode = CDATA**

This attribute specifies style information for the current element.

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

<table>
<thead>
<tr>
<th>Elements</th>
<th>Attributes</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>inputmode (CDATA)</td>
<td>When the Basic Forms or Forms Module is selected.</td>
</tr>
<tr>
<td>textarea</td>
<td>inputmode (CDATA)</td>
<td>When the Basic Forms or Forms Module is selected.</td>
</tr>
</tbody>
</table>

The attribute inputmode provides a hint to the user agent to select an appropriate input mode for the text input expected in an associated form control. The input mode may be a keyboard configuration, an input method editor (also called front end processor) or any other setting affecting input on the device(s) used.

Using inputmode, the author can give hints to the agent that make form input easier for the user. Authors should provide inputmode attributes wherever possible, making sure that the values used cover a wide range of devices.

### 5.1 inputmode Attribute Value Syntax

The value of the inputmode attribute is a white space separated list of tokens. Tokens are either sequences of alphabetic letters or absolute URIs. The latter can be distinguished from the former by noting that absolute URIs contain a ‘:’. Tokens are case-sensitive. All the tokens consisting of alphabetic letters only are defined in this specification, in [5.3 List of Tokens][17] (or a successor of this specification).

This specification does not define any URIs for use as tokens, but allows others to define such URIs for extensibility. This may become necessary for devices with input modes that cannot be covered by the tokens provided here. The URI should dereference to a human-readable description of the input mode associated with the use of the URI as a token. This description should describe the input mode indicated by this token, and whether and how this token modifies other tokens or is modified by other tokens.
5.2 User Agent Behavior

Upon entering an empty form control with an inputmode attribute, the user agent should select the input mode indicated by the inputmode attribute value. User agents should not use the inputmode attribute to set the input mode when entering a form control with text already present. To set the appropriate input mode when entering a form control that already contains text, user agents should rely on platform-specific conventions.

User agents should make available all the input modes which are supported by the (operating) system/device(s) they run on/have access to, and which are installed for regular use by the user. This is typically only a small subset of the input modes that can be described with the tokens defined here.

Note:

Additional guidelines for user agent implementation are found at [UAAG 1.0][p.24].

The following simple algorithm is used to define how user agents match the values of an inputmode attribute to the input modes they can provide. This algorithm does not have to be implemented directly; user agents just have to behave as if they used it. The algorithm is not designed to produce "obvious" or "desirable" results for every possible combination of tokens, but to produce correct behavior for frequent token combinations and predictable behavior in all cases.

First, each of the input modes available is represented by one or more lists of tokens. An input mode may correspond to more than one list of tokens; as an example, on a system set up for a Greek user, both "greek upperCase" and "user upperCase" would correspond to the same input mode. No two lists will be the same.

Second, the inputmode attribute is scanned from front to back. For each token $t$ in the inputmode attribute, if in the remaining lists of tokens representing available input modes there is any list of tokens that contains $t$, then all lists of tokens representing available input modes that do not contain $t$ are removed. If there is no remaining list of tokens that contains $t$, then $t$ is ignored.

Third, if one or more lists of tokens are left, and they all correspond to the same input mode, then this input mode is chosen. If no list is left (meaning that there was none at the start) or if the remaining lists correspond to more than one input mode, then no input mode is chosen.

Example: Assume the list of lists of tokens representing the available input modes is: {"cyrillic upperCase", "cyrillic lowerCase", "cyrillic", "latin", "user upperCase", "user lowerCase"}, then the following inputmode values select the following input modes: "cyrillic title" selects "cyrillic", "cyrillic lowerCase" selects "cyrillic lowerCase", "lowerCase cyrillic" selects "cyrillic lowerCase", "latin upperCase" selects "latin", but "uppercase latin" does select "cyrillic upperCase" or "user upperCase" if they correspond to the same input mode, and does not select any input mode if "cyrillic upperCase" and "user upperCase" do not correspond to the same input mode.
5.3 List of Tokens

Tokens defined in this specification are separated into two categories: Script tokens and modifiers. In inputmode attributes, script tokens should always be listed before modifiers.

5.3.1 Script Tokens

Script tokens provide a general indication the set of characters that is covered by an input mode. In most cases, script tokens correspond directly to [Unicode Scripts][p.24]. Some tokens correspond to the block names in Java class java.lang.Character.UnicodeBlock ([Java Unicode Blocks][p.24]) or Unicode Block names. However, this neither means that an input mode has to allow input for all the characters in the script or block, nor that an input mode is limited to only characters from that specific script. As an example, a "latin" keyboard doesn’t cover all the characters in the Latin script, and includes punctuation which is not assigned to the Latin script. The version of the Unicode Standard that these script names are taken from is 3.2.

<table>
<thead>
<tr>
<th>Input Mode Token</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>arabic</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>armenian</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>bengali</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>bopomofo</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>braille</td>
<td>used to input braille patterns (not to indicate a braille input device)</td>
</tr>
<tr>
<td>buhid</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>canadianAboriginal</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>cherokee</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>cyrillic</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>deseret</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>devanagari</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>ethiopic</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>georgian</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>greek</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>gothic</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>gujarati</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>gurmukhi</td>
<td>Unicode script name</td>
</tr>
<tr>
<td><strong>Input Mode Token</strong></td>
<td><strong>Comments</strong></td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>han</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>hangul</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>hanja</td>
<td>Subset of 'han' used in writing Korean</td>
</tr>
<tr>
<td>hanunoo</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>hebrew</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>hiragana</td>
<td>Unicode script name (may include other Japanese scripts produced by conversion from hiragana)</td>
</tr>
<tr>
<td>ipa</td>
<td>International Phonetic Alphabet</td>
</tr>
<tr>
<td>kanji</td>
<td>Subset of 'han' used in writing Japanese</td>
</tr>
<tr>
<td>kannada</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>katakana</td>
<td>Unicode script name (full-width, not half-width)</td>
</tr>
<tr>
<td>khmer</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>lao</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>latin</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>malayalam</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>math</td>
<td>mathematical symbols and related characters</td>
</tr>
<tr>
<td>mongolian</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>myanmar</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>ogham</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>oldItalic</td>
<td>Unico de script name</td>
</tr>
<tr>
<td>oriya</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>runic</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>simplifiedHanzi</td>
<td>Subset of 'han' used in writing Simplified Chinese</td>
</tr>
<tr>
<td>sinhala</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>syriac</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>tagalog</td>
<td>Unicode script name</td>
</tr>
<tr>
<td>tagbanwa</td>
<td>Unicode script name</td>
</tr>
</tbody>
</table>
5.3.2 Modifier Tokens

Modifier tokens can be added to the scripts they apply in order to more closely specify the kind of characters expected in the form control. Traditional PC keyboards do not need most modifier tokens (indeed, users on such devices would be quite confused if the software decided to change case on its own; CAPS lock for upperCase may be an exception). However, modifier tokens can be very helpful to set input modes for small devices.

<table>
<thead>
<tr>
<th>Input Mode Token</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>lowerCase</td>
<td>lowercase (for bicameral scripts)</td>
</tr>
<tr>
<td>upperCase</td>
<td>uppercase (for bicameral scripts)</td>
</tr>
<tr>
<td>titleCase</td>
<td>title case (for bicameral scripts): words start with an upper case letter</td>
</tr>
<tr>
<td>startUpper</td>
<td>start input with one uppercase letter, then continue with lowercase letters</td>
</tr>
<tr>
<td>digits</td>
<td>digits of a particular script (e.g. inputmode='thai digits')</td>
</tr>
<tr>
<td>symbols</td>
<td>symbols, punctuation (suitable for a particular script)</td>
</tr>
<tr>
<td>predictOn</td>
<td>text prediction switched on (e.g. for running text)</td>
</tr>
<tr>
<td>predictOff</td>
<td>text prediction switched off (e.g. for passwords)</td>
</tr>
<tr>
<td>halfWidth</td>
<td>half-width compatibility forms (e.g. Katakana; deprecated)</td>
</tr>
</tbody>
</table>
5.4 Relationship to XML Schema pattern facets

User agents may use information available in an XML Schema pattern facet to set the input mode. Note that a pattern facet is a hard restriction on the lexical value of an instance data node, and can specify different restrictions for different parts of the data item. Attribute inputmode is a soft hint about the kinds of characters that the user may most probably start to input into the form control. Attribute inputmode is provided in addition to pattern facets for the following reasons:

1. The set of allowable characters specified in a pattern may be so wide that it is not possible to deduce a reasonable input mode setting. Nevertheless, there frequently is a kind of characters that will be input by the user with high probability. In such a case, inputmode allows to set the input mode for the user’s convenience.

2. In some cases, it would be possible to derive the input mode setting from the pattern because the set of characters allowed in the pattern closely corresponds to a set of characters covered by an inputmode attribute value. However, such a derivation would require a lot of data and calculations on the user agent.

3. Small devices may leave the checking of patterns to the server, but will easily be able to switch to those input modes that they support. Being able to make data entry for the user easier is of particular importance on small devices.

5.5 Examples

This is an example of a form for Japanese address input.

```xml
Family name: <input name="name" inputmode="kanji" />
(in kana):   <input name="namekana" inputmode="katakana" />
Given name:  <input name="Given" inputmode="kanji" />
(in kana):   <input name="Givenkana" inputmode="katakana" />
Postal code: <input name="zip" inputmode="latin digits" />
Address:     <input name="address" inputmode="kanji" />
(in kana):   <input name="addresskana" inputmode="katakana" />
Email:       <input name="email" inputmode="latin lowerCase" />
Telephone:   <input name="tel" inputmode="latin digits" />
Comments:    <textarea name="comments" inputmode="user predictOn" />
<input name="doit" type="submit" value="OK" />
```
6. Acknowledgements

Version 1.0 of this specification was prepared by the W3C HTML Working Group. At the time of publication of the first edition, the members were:

- Steven Pemberton, CWI (HTML Working Group Chair)
- Robert Adams, Intel (until November 2000)
- Murray Altheim, Sun Microsystems
- Takuya Asada, W3C (until October 2000)
- Daniel Austin, Mozquito Technologies
- Mark Baker, Sun Microsystems
- Wayne Carr, Intel
- Tantek Çelik, Microsoft
- Andrew W. Donoho, IBM
- Herman Elenbaas, Philips Electronics
- Beth Epperson, Netscape/AOL
- Masayasu Ishikawa, W3C (HTML Activity Lead)
- Shin’ichi Matsui, Panasonic
- Shane McCarron, Applied Testing and Technology
- Ann Navarro, WebGeek, Inc.
- Dave Raggett, W3C/Openwave Systems
- Sebastian Schnitzenbaumer, Mozquito Technologies (until September 2000)
- Peter Stark, Ericsson
- Michel Suignard, Microsoft
- Markku Vartiainen, Openwave Systems
- Jeremy Wadsworth, Quark Inc.
- Malte Wedel, Mozquito Technologies
- Linda Welsh, Intel
- Ted Wugofski, Openwave Systems

Version 1.1 of this specification was produced by the W3C XHTML2 Working Group. At the time of publication, the members were:

- Mark Birbeck, XPort.net
- Susan Borgrink, Progeny Systems
- Alessio Cartocci, International Webmasters Association / HTML Writers Guild (IWA-HWG)
- Alexander Graf, University of Innsbruck
- Tina Holmboe, Greytower
- John Kugelman, Progeny Systems
- Luca Mascaro, International Webmasters Association / HTML Writers Guild (IWA-HWG)
- Shane McCarron, Applied Testing and Technology
- Roland Merrick (chair), IBM Corporation
- Steven Pemberton (chair, staff contact), CWI and W3C
- Michael Rawling, Ivis Group Limited
6. Acknowledgements

- Sebastian Schnitzenbaumer, Dreamlab Technologies AG
- Richard Schwerdtfeger, IBM Corporation
- Elias Torres, IBM Corporation
- Masataka Yakura, Mitsue-Links Co., Ltd.
- Toshihiko Yamakami, ACCESS Co., Ltd.

At publication of the second edition, the membership was:

- Roland Merrick, IBM (XHTML 2 Working Group Co-Chair)
- Steven Pemberton, CWI (XHTML 2 Working Group Co-Chair)
- Mark Birbeck, webBackplane (Invited Expert)
- Susan Borgrink, Progeny Systems
- Christina Bottomley, Society for Technical Communication (STC)
- Alessio Cartocci, International Webmasters Association / HTML Writers Guild (IWA-HWG)
- Alexander Graf, University of Innsbruck
- Markus Gylling, DAISY Consortium
- Tina Holmboe, Greytower Technologies (Invited Expert)
- John Kugelman, Progeny Systems
- Luca Mascaro, International Webmasters Association / HTML Writers Guild (IWA-HWG)
- Shane McCarron, Applied Testing and Technology, Inc. (Invited Expert)
- Michael Rawling, IVIS Group Limited
- Gregory Rosmaita, Invited Expert
- Sebastian Schnitzenbaumer, Dreamlab Technologies AG
- Richard Schwerdtfeger, IBM
- Elias Torres, IBM
- Masataka Yakura, Mitsue-Links Co., Ltd.
- Toshihiko Yamakami, ACCESS Co., Ltd.

Thanks to Gary Adams (Sun Microsystems), Jonny Axelsson (Metastasis design), Peter Chen (Philips), Dan Connolly (W3C), John Cowan (Reuters), Martin J. Dürst (W3C), Johan Hjelm (Ericsson), Ian Jacobs (W3C), Susan Lesch (W3C), Louis Theran (Nokia), Quinton Zondervan (Lotus), members of the W3C Mobile Access Interest Group, the W3C Synchronized Multimedia Working Group, the W3C WAI Protocols and Formats Working Group, and the Open Mobile Alliance, for contributing, reviewing and commenting on this document.
A. References

A.1. Normative References

[HTML4]


The latest version is available at: http://www.w3.org/TR/html4

[RFC2854]


[RFC3236]


[XHTML1]

"XHTML 1.0: The Extensible HyperText Markup Language (Second Edition) - A Reformulation of HTML 4 in XML 1.0", W3C Recommendation, Steven Pemberton et al., 26 January 2000, revised 1 August 2002. Available at: http://www.w3.org/TR/2002/REC-xhtml1-20020801

The latest version is available at: http://www.w3.org/TR/xhtml1

[XHTMLMOD]


The latest version is available at: http://www.w3.org/TR/xhtml-modularization

[XML]

"Extensible Markup Language (XML) 1.0 (Fourth Edition)", W3C Recommendation, T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau, eds., 16 August 2006. Available at: http://www.w3.org/TR/2006/REC-xml-20060816

The latest version is available at: http://www.w3.org/TR/REC-xml

[XMLSCHEMA]


A.2. Informative References

[CHTML]


[GUIDELINES]

Java Unicode Blocks

Java 2 Platform, Standard Edition, v 1.4.0 API Specification; Class Character.UnicodeBlock

Mobile Web Best Practices

The latest version is available at: http://www.w3.org/TR/mobile-bp

UAAG 1.0


Unicode Scripts


[WCAG10]

The latest version is available at: http://www.w3.org/TR/WCAG10

[WML]

"Wireless Markup Language Specification", WAP Forum Ltd. The WAP Forum has consolidated into the Open Mobile Alliance (OMA). The specification work from WAP continues within OMA and can be found on the OMA Web site at: http://www.openmobilealliance.org/tech/affiliates/wap/wapindex.html

[XFORMS]


[XHTMLMIME]

[XHTML Media Types], Shane McCarron, 16 January 2009, or its successors. Latest version available at: http://www.w3.org/TR/xhtml-media-types
B. XHTML Basic Document Type Definition

This appendix is normative.

The DTD Implementation of XHTML Basic 1.1 is contained in this appendix. There are direct links to the various files, and the files are also contained in the "Gzip'd TAR" and "Zip" archives linked to at the top of this document. Please note that the files targeted by the "latest version" links may change slowly over time. See the W3C XHTML2 Working Group home page for more information.

B.1. SGML Open Catalog Entry for XHTML Basic

This section contains the SGML Open Catalog-format definition of the public identifiers for XHTML Basic.

You can download this version of this file from http://www.w3.org/TR/2009/PER-xhtml-basic-20101007/xhtml-basic11.cat. The latest version is available at http://www.w3.org/MarkUp/DTD/xhtml-basic11.cat.

```
-- .......................................................................... --
-- File catalog ............................................................ --
-- XHTML Basic Catalog Data File
Revision: $Id: xhtml-basic11.cat,v 1.1.2.2 2006/06/28 18:15:52 ahby Exp $ SMI

See "Entity Management", SGML Open Technical Resolution 9401 for detailed information on supplying and using catalog data. This document is available from OASIS at URL:


--
-- .......................................................................... --
-- SGML declaration associated with XML .................................... --

OVERRIDE YES

SGMLDECL "xml1.dcl"
-- :::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: --
-- XHTML Basic DTD modular driver file ..................................... --
PUBLIC "-//W3C//DTD XHTML Basic 1.1//EN" "xhtml-basic11.dtd"
-- XHTML Basic framework module ............................................. --
PUBLIC "-//W3C//ENTITIES XHTML Basic 1.1 Document Model 1.0//EN" "xhtml-basic11-model-1.mod"
-- XHTML Inputmode module ..................................................... --
PUBLIC "-//W3C//ELEMENTS XHTML Inputmode 1.0//EN" "xhtml-inputmode-1.mod"
-- End of catalog data ............................................................. --
-- .......................................................................... --
```
B.2. XHTML Basic Driver

This section contains the driver for the XHTML Basic document type implementation as an XML DTD. It relies upon XHTML module implementations defined in [XHTMLMOD][p.23].

You can download this version of this file from http://www.w3.org/TR/2009/PER-xhtml-basic-20101007/xhtml-basic11.dtd. The latest version is available at http://www.w3.org/MarkUp/DTD/xhtml-basic11.dtd.

<!-- XHTML Basic 1.1 DTD ...................................................... -->
<!-- file: xhtml-basic11.dtd -->

<!-- XHTML Basic 1.1 DTD ...................................................... -->
<!-- file: xhtml-basic11.dtd -->

<!-- XHTML Basic 1.1 DTD
This is XHTML Basic, a proper subset of XHTML.

The Extensible HyperText Markup Language (XHTML)
Copyright 1998-2007 World Wide Web Consortium
(Massachusetts Institute of Technology, European Research Consortium
for Informatics and Mathematics, Keio University).
All Rights Reserved.

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

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

Editors: Murray M. Altheim <mailto:altheim@eng.sun.com>
Peter Stark <mailto:Peter.Stark@ecs.ericsson.se>
Shane McCarron <mailto:shane@aptest.com>

<!-- This is the driver file for version 1.1 of the XHTML Basic DTD.

This DTD is identified by the PUBLIC and SYSTEM identifiers:

PUBLIC: "-//W3C//DTD XHTML Basic 1.1//EN"
SYSTEM: "http://www.w3.org/MarkUp/DTD/xhtml-basic11.dtd"

<!ENTITY % XHTML.version "-//W3C//DTD XHTML Basic 1.1//EN">

<!-- Use this URI to identify the default namespace:

"http://www.w3.org/1999/xhtml"

See the Qualified Names module for information
on the use of namespace prefixes in the DTD.

<!ENTITY % NS.prefixed "IGNORE" >
<!ENTITY % XHTML.prefix "" >
<!-- Reserved for use with the XLink namespace: -->
<!ENTITY % XLINK.xmlns "" >
<!ENTITY % XLINK.xmlns.attrib "" >

<!-- For example, if you are using XHTML Basic 1.1 directly, use the public identifier in the DOCTYPE declaration, with the namespace declaration on the document element to identify the default namespace: -->

<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.1//EN" "http://www.w3.org/MarkUp/DTD/xhtml-basic11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
  ...
</html>

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

<!-- Bidirectional Text features
This feature-test entity is used to declare elements and attributes used for bidirectional text support. -->
<!ENTITY % XHTML.bidi "IGNORE" >

<?doc type="doctype" role="title" { XHTML Basic 1.1 } ?>

<!-- :..........................doctype" role="title" { XHTML Basic 1.1 } -->

<!ENTITY % xhtml-events.module "INCLUDE" >
<!ENTITY % xhtml-bdo.module "%XHTML.bidi;" >

<!-- Inline Style Module ........................................ -->
<!ENTITY % xhtml-inlstyle.module "INCLUDE" >
<!ENTITY % xhtml-inlstyle.module;[
<!ENTITY % xhtml-inlstyle.mod PUBLIC "-//W3C//ELEMENTS XHTML Inline Style 1.0//EN" "http://www.w3.org/MarkUp/DTD/xhtml-inlstyle-1.mod" >
%xhtml-inlstyle.mod;]]>
<!ENTITY % xhtml-model.mod PUBLIC "-//W3C//ENTITIES XHTML Basic 1.1 Document Model 1.0//EN" "http://www.w3.org/MarkUp/DTD/xhtml-basic11-model-1.mod" >

<!ENTITY % xhtml-model.mod.Public "-//W3C//ENTITIES XHTML Basic 1.1 Document Model 1.0//EN" "http://www.w3.org/MarkUp/DTD/xhtml-basic11-model-1.mod" >

<!-- adding the lang attribute into the I18N collection -->
<!ENTITY % xhtml-datatypes.module "INCLUDE" >
<!ENTITY % xhtml-datatypes.module;[
<!ENTITY % xhtml-datatypes.mod PUBLIC "-//W3C//ENTITIES XHTML Datatypes 1.0//EN" "xhtml-datatypes-1.mod" >
%xhtml-datatypes.mod;]]>

<!ENTITY % lang.attrib
"xml:lang     %LanguageCode.datatype;   #IMPLIED
  lang         %LanguageCode.datatype;   #IMPLIED"
>
<!ENTITY % xhtml-framework.mod
  PUBLIC "-//W3C//ENTITIES XHTML Modular Framework 1.0//EN"
  "http://www.w3.org/MarkUp/DTD/xhtml-framework-1.mod" >
%xhtml-framework.mod;

<!ENTITY % pre.content
  "(#PCDATA
  | %InlStruct.class;
  %InlPhras.class;
  %Anchor.class;
  %Inline.extra; )""
>
<!ENTITY % xhtml-text.mod
  PUBLIC "-//W3C//ELEMENTS XHTML Text 1.0//EN"
  "http://www.w3.org/MarkUp/DTD/xhtml-text-1.mod" >
%xhtml-text.mod;

<!ENTITY % xhtml-hypertext.mod
  PUBLIC "-//W3C//ELEMENTS XHTML Hypertext 1.0//EN"
  "http://www.w3.org/MarkUp/DTD/xhtml-hypertext-1.mod" >
%xhtml-hypertext.mod;

<!ENTITY % xhtml-list.mod
  PUBLIC "-//W3C//ELEMENTS XHTML Lists 1.0//EN"
  "http://www.w3.org/MarkUp/DTD/xhtml-list-1.mod" >
%xhtml-list.mod;

<!-- Add in the value attribute to the li element -->
<!ATTLIST %li.qname
  value      %Number.datatype;          #IMPLIED
>
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
<!-- Scripting Module  ........................................... -->
<!ENTITY % xhtml-script.module "INCLUDE" >
<![%xhtml-script.module;[
<!ENTITY % xhtml-script.mod
  PUBLIC "-//W3C//ELEMENTS XHTML Scripting 1.0//EN"
  "http://www.w3.org/MarkUp/DTD/xhtml-script-1.mod" >
%xhtml-script.mod;]]>

<!-- Style Sheets Module  ......................................... -->
<!ENTITY % xhtml-style.module "INCLUDE" >
<![%xhtml-style.module;[
<!ENTITY % xhtml-style.mod
  PUBLIC "-//W3C//ELEMENTS XHTML Style Sheets 1.0//EN"
  "http://www.w3.org/MarkUp/DTD/xhtml-style-1.mod" >
%xhtml-style.mod;]]>

<!-- Image Module  ............................................... -->
<!ENTITY % xhtml-image.module "INCLUDE" >

- 28 -
B.3. XHTML Basic Customizations

An XHTML Family Document Type (such as XHTML Basic) must define the content model that it uses. This is done through a separate content model module that is instantiated by the XHTML Modular Framework. The content model module and the XHTML Basic Driver (above) work together to customize the module implementations to the document type's specific requirements. The content model module for XHTML Basic is defined below:

You can download this version of this file from http://www.w3.org/TR/2009/PER-xhtml-basic-20101007/xhtml-basic11-model-1.mod. The latest version is available at http://www.w3.org/MarkUp/DTD/xhtml-basic11-model-1.mod.

<!DOCTYPE % xhtml-param.mod
PUBLIC "-//W3C//ELEMENTS XHTML Param Element 1.0//EN"
"http://www.w3.org/MarkUp/DTD/xhtml-param-1.mod" >
%xhtml-param.mod;]]>

<!-- Embedded Object Module ................................. -->
<!ENTITY % xhtml-object.module "INCLUDE" >
<![%xhtml-object.module;[
<!ENTITY % xhtml-object.mod
PUBLIC "-//W3C//ELEMENTS XHTML Embedded Object 1.0//EN"
"http://www.w3.org/MarkUp/DTD/xhtml-object-1.mod" >
%xhtml-object.mod;]]>

<!-- Inputmode Attribute Module ............................ -->
<!ENTITY % xhtml-inputmode.module "INCLUDE" >
<![%xhtml-inputmode.module;[
<!ENTITY % xhtml-inputmode.mod
PUBLIC "-//W3C//ELEMENTS XHTML Inputmode 1.0//EN"
"http://www.w3.org/MarkUp/DTD/xhtml-inputmode-1.mod" >
%xhtml-inputmode.mod;]]>

<!-- Target Attribute Module ............................... -->
<!ENTITY % xhtml-target.module "INCLUDE" >
<![%xhtml-target.module;[
<!ENTITY % xhtml-target.mod
PUBLIC "-//W3C//ELEMENTS XHTML Target 1.0//EN"
"http://www.w3.org/MarkUp/DTD/xhtml-target-1.mod" >
%xhtml-target.mod;]]>

<!ENTITY % xhtml-struct.mod
PUBLIC "-//W3C//ELEMENTS XHTML Document Structure 1.0//EN"
"http://www.w3.org/MarkUp/DTD/xhtml-struct-1.mod" >
%xhtml-struct.mod;

<!-- end of XHTML Basic 1.1 DTD .............................. -->

This is XHTML Basic, a proper subset of XHTML.
Copyright 1998-2007 W3C (MIT, ERCIM, Keio), All Rights Reserved.
This DTD module is identified by the PUBLIC and SYSTEM identifiers:

PUBLIC "-//W3C//ENTITIES XHTML Basic 1.1 Document Model 1.0//EN"
SYSTEM "http://www.w3.org/MarkUp/DTD/xhtml-basic11-model-1.mod"

Revisions:
(none)

-->

<!-- XHTML Basic Document Model

This module describes the groupings of elements that make up
common content models for XHTML elements.
-->

<!-- Optional Elements in head .............. -->

<!-- script and noscript are used to contain scripts
and alternative content
-->

<!ENTITY % HeadOpts.mix "( %script.qname; | %style.qname; | %meta.qname;
| %link.qname; | %object.qname; )**" >

<!ENTITY % Script.class "| %script.qname; | %noscript.qname;" >

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

<!ENTITY % Misc.class "%Script.class;
%Misc.extra;" >

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

<!ENTITY % Misc.extra "" >

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

<!ENTITY % InlStruct.class "%br.qname; | %span.qname;" >

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

<!ENTITY % InlPres.class "| %tt.qname; | %i.qname; | %b.qname; | %big.qname;
| %small.qname; | %sub.qname; | %sup.qname;" >

<!ENTITY % I18n.class "" >

<!ENTITY % Anchor.class "| %a.qname;" >

<!ENTITY % InlSpecial.class "| %img.qname; | %object.qname;" >

<!ENTITY % InlForm.class "| %input.qname; | %select.qname; | %textarea.qname;
| %label.qname; | %button.qname;" >
Finally, we define the new inputmode attribute module.

You can download this version of this file from http://www.w3.org/TR/2009/PER-xhtml-basic-20101007/xhtml-inputmode-1.mod. The latest version is available at http://www.w3.org/MarkUp/DTD/xhtml-inputmode-1.mod.
Revisions:
(none)

This module declares the 'inputmode' attribute used for suggesting the input mode associated with an input or textarea element.

<!-- Inputmode

inputmode

    This module declares the 'inputmode' attribute used for suggesting the input mode associated with an input or textarea element.

-->

<!ENTITY % Inputmode.datatype "CDATA" >

<!-- add 'inputmode' attribute to 'input' element -->
<!ATTLIST %input.qname;
    inputmode %Inputmode.datatype;   #IMPLIED
>

<!-- add 'inputmode' attribute to 'textarea' element -->
<!ATTLIST %textarea.qname;
    inputmode %Inputmode.datatype;   #IMPLIED
>

<!-- end of xhtml-inputmode-1.mod -->
C. XHTML Basic XML Schema Definition

This appendix is **normative**.

C.1. XHTML Basic XML Schema Driver

This section contains the driver for the XHTML Basic document type implementation as an XML Schema. It relies upon XHTML module implementations defined in [XHTMLMOD][p.23].

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.w3.org/1999/xhtml"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
xmlns="http://www.w3.org/1999/xhtml"
blockDefault="#all">
<xs:annotation>
<xs:documentation>
This is the XML Schema driver for XHTML Basic 1.1.
Please use this namespace for XHTML elements:
"http://www.w3.org/1999/xhtml"

$Id: xhtml-basic11.xsd,v 1.1.2.1 2008/05/29 17:31:18 ahby Exp $
</xs:documentation>
</xs:annotation>

<xs:annotation>
<xs:documentation>
This is XHTML Basic
Copyright ©1998-2008 World Wide Web Consortium
(Massachusetts Institute of Technology, European Research Consortium
for Informatics and Mathematics, Keio University). All Rights Reserved.

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

They are provided "as is" without expressed or implied warranty.
</xs:documentation>
</xs:annotation>

<xs:annotation>
<xs:documentation>
This is the Schema Driver file for XHTML Basic 1.1
Document Type

This schema includes
+ imports external schemas (xml.xsd)
+ redefines (and include)s schema modules for XHTML
  Basic 1.1 Document Type.
+ includes Schema for Named content model for the
```
XHTML Basic 1.1 Document Type

XHTML Basic 1.1 Document Type includes the following Modules

XHTML Core modules (Required for XHTML Family Conformance)
+ text
+ hypertext
+ lists
+ structure (redefined)

Other XHTML modules
+ Link
+ Metainformation
+ Intrinsic Events
+ Scripting
+ Stylesheet
+ Style Attribute
+ Target
+ Inputmode
+ Base
+ Image
+ Object
+ Presentation
+ Param
+ Forms
+ Basic tables

</xs:documentation>
</xs:annotation>

schemaLocation="http://www.w3.org/2001/xml.xsd">
<xs:annotation>
<xs:documentation>
This import brings in the XML namespace attributes
The XML attributes are used by various modules
</xs:documentation>
</xs:annotation>
</xs:import>

<xs:include schemaLocation="xhtml-basic11-model-1.xsd">
<xs:annotation>
<xs:documentation>
Document Model module for the XHTML Basic 1.1 Document Type
This schema file defines all named models used by XHTML Modularization Framework for XHTML Basic 1.1 Document Type
</xs:documentation>
</xs:annotation>
</xs:include>

<xs:include schemaLocation="xhtml-basic11-modules-1.xsd">
<xs:annotation>
<xs:documentation>
Schema that includes the modules (and redefinitions)
for XHTML Basic 1.1 Document Type.
</xs:documentation>
</xs:annotation>
</xs:include>
C.2. XHTML Basic Schema Modules

XHTML Family implementations using XML Schema are required to provide their own schema module that imports the required modules from XHTML Modularization.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
        schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-datatypes-1.xsd" />
```

This schema includes all modules for XHTML Basic 1.1 Document Type.

Other XHTML modules
- Link
- Meta
- Base
- Image
- Object
- Param
- Basic forms
- Basic tables

```xml
<xs:include schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-framework-1.xsd"/>
```

Schema Framework Component Modules:
- notations
- datatypes
- common attributes
- character entities

```xml
<xs:include schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-text-1.xsd"/>
```

Text module

Elements defined here:
- address, blockquote, pre, h1, h2, h3, h4, h5, h6
- div, p
- abbr, acronym, cite, code, dfn, em, kbd, q, samp, strong, var
- br, span

```xml
<xs:include schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-hypertext-1.xsd"/>
```

Hypertext module
Elements defined here:
* a
</xs:documentation>
</xs:annotation>

<xs:attributeGroup name="xhtml.a.attlist">
<xs:attributeGroup ref="xhtml.a.events.attlist"/>
<xs:attributeGroup ref="xhtml.a.target.attlist"/>
<xs:annotation>
<xs:documentation>
Redefinition by XHTML Event Attribute Module
</xs:documentation>
</xs:attributeGroup>
</xs:attributeGroup>
</xs:redefine>

<xs:redefine schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-list-1.xsd">
<xs:annotation>
<xs:documentation>
Lists module
</xs:documentation>
</xs:annotation>
</xs:redefine>

<xs:redefine schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-struct-1.xsd">
<xs:annotation>
<xs:documentation>
Structural module
</xs:documentation>
</xs:annotation>
</xs:redefine>

<xs:redefine schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-pres-1.xsd">
<xs:annotation>
<xs:documentation>
Presentational module
</xs:documentation>
</xs:annotation>
</xs:redefine>

<xs:redefine schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-link-1.xsd">
<xs:annotation>
<xs:documentation>
Link module
</xs:documentation>
</xs:annotation>
</xs:redefine>
Changes to XHTML Link Attlist

Original Link Attributes (declared in Link Module)

XHTML Target Module - Attribute additions

Changes to XHTML base Attlist

Original Base Attributes (declared in Base Module)

XHTML Target Module - Attribute additions

Meta module

Base module

Scripting module

Style module

Style attribute module
C.2. XHTML Basic Schema Modules

The document contains references to the Image Module, Object Module, Param Module, and Form Module. Each module is defined with references to the relevant XML schema locations and documentation sources. The Image Module includes elements such as `img`, and the Object Module includes elements such as `object`. The Param Module includes elements such as `param`, and the Form Module includes elements such as `form`, `label`, `input`, `select`, `optgroup`, `option`, `textarea`, `fieldset`, `legend`, `button`. The Basic Tables module is noted to be used with XHTML Basic and includes elements such as `table`, `caption`, `tr`, `th`, `td`. The Forms module includes changes to the XHTML Form Attlist and attributes such as `form`, `label`, `input`, `select`, `optgroup`, `option`, `textarea`, `fieldset`, `legend`, `button`. The XHTML Events Module includes attribute additions.
C.3. XHTML Basic Customizations

An XHTML Family Document Type (such as XHTML Basic) must define the content model that it uses. This is done through a separate content model module that is instantiated by the XHTML Modular Framework. The content model module and the XHTML Basic Driver (above) work together to customize the module implementations to the document type’s specific requirements. The content model module for XHTML Basic is defined below:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
    xmlns:xhtml="http://www.w3.org/1999/xhtml"
    <xs:import namespace="http://www.w3.org/1999/xhtml/datatypes/"
        schemaLocation="http://www.w3.org/MarkUp/SCHEMA/xhtml-datatypes-1.xsd"/>
    <xs:annotation>
        <xs:documentation>
            This is the XML Schema module of named XHTML content models for XHTML Basic 1.0
        </xs:documentation>
    </xs:annotation>
</xs:schema>
```

C.3. Extensible HyperText Markup Language XHTML Basic Customizations
This module describes the groupings of elements/attributes that make up common content models for XHTML elements. XHTML has following basic content models:
xhtml.Inline.mix; character-level elements
xhtml.Block.mix; block-like elements, e.g., paragraphs and lists
xhtml.Flow.mix; any block or inline elements
xhtml.HeadOpts.mix; Head Elements
xhtml.InlinePre.mix; Special class for pre content model
xhtml.InlineNoAnchor.mix; Content model for Anchor

Any groups declared in this module may be used to create element content models, but the above are considered 'global' (insofar as that term applies here). XHTML has the following Attribute Groups
xhtml.Core.extra.attrib
xhtml.I18n.extra.attrib
xhtml.Common.extra

The above attribute Groups are considered Global
<xs:attributeGroup
    name="xhtml.Global.core.extra.attrib">
  <xs:annotation>
    <xs:documentation> Extended Global Core Attributes </xs:documentation>
  </xs:annotation>
</xs:attributeGroup>
<xs:attributeGroup
    name="xhtml.Global.I18n.extra.attrib">
  <xs:annotation>
    <xs:documentation> Extended Global I18n attributes </xs:documentation>
  </xs:annotation>
</xs:attributeGroup>
<xs:attributeGroup
    name="xhtml.Global.Common.extra">
  <xs:annotation>
    <xs:documentation> Extended Global Common Attributes </xs:documentation>
  </xs:annotation>
</xs:attributeGroup>
<xs:group
    name="xhtml.Head.extra">
  <xs:sequence/>
</xs:group>
<xs:group
    name="xhtml.HeadOpts.mix">
  <xs:choice>
    <xs:element
        name="script"
        type="xhtml.script.type"/>
    <xs:element
        name="style"
        type="xhtml.style.type"/>
    <xs:element
        name="meta"
        type="xhtml.meta.type"/>
    <xs:element
        name="link"
        type="xhtml.link.type"/>
    <xs:element
        name="object"
        type="xhtml.object.type"/>
    <xs:group
        ref="xhtml.Head.extra"/>
  </xs:choice>
</xs:group>
<xs:group
    name="xhtml.head.content">
  <xs:sequence>
    <xs:group
        ref="xhtml.HeadOpts.mix"
        minOccurs="0"
        maxOccurs="unbounded"/>
    <xs:choice>
      <xs:sequence>
        <xs:element
            name="title"
            minOccurs="1"
            maxOccurs="1"/>
<xs:group name="xhtml.Script.class">
  <xs:choice>
    <xs:element name="script" type="xhtml.script.type"/>
    <xs:element name="noscript" type="xhtml.noscript.type"/>
  </xs:choice>
</xs:group>

<xsl:group name="xhtml.Misc.extra">
  <xs:sequence/>
</xsl:group>

<!--script and noscript are used to contain scripts and alternative content -->

These elements are neither block nor inline, and can essentially be used anywhere in the document body.

<!-- Ini Elements -->

<xs:group name="xhtml.InlStruct.class">
  <xs:choice>
    <xs:element name="br" type="xhtml.br.type"/>
    <xs:element name="span" type="xhtml.span.type"/>
  </xs:choice>
</xs:group>

<xs:group name="xhtml.InlPhras.class">
  <xs:choice>
    <xs:element name="em" type="xhtml.em.type"/>
    <xs:element name="strong" type="xhtml.strong.type"/>
    <xs:element name="dfn" type="xhtml.dfn.type"/>
    <xs:element name="code" type="xhtml.code.type"/>
    <xs:element name="samp" type="xhtml.samp.type"/>
    <xs:element name="kbd" type="xhtml.kbd.type"/>
    <xs:element name="var" type="xhtml.var.type"/>
    <xs:element name="cite" type="xhtml.cite.type"/>
    <xs:element name="abbr" type="xhtml.abbr.type"/>
    <xs:element
name="acronym"
type="xhtml.acronym.type"/>
</xs:choice>
</xs:group>
<xs:group
name="xhtml.InlPres.class">
<xs:choice>
<xs:element
name="tt"
type="xhtml.InlPres.type"/>
<xs:element
name="i"
type="xhtml.InlPres.type"/>
<xs:element
name="b"
type="xhtml.InlPres.type"/>
<xs:element
name="big"
type="xhtml.InlPres.type"/>
<xs:element
name="small"
type="xhtml.InlPres.type"/>
<xs:element
name="sub"
type="xhtml.InlPres.type"/>
<xs:element
name="sup"
type="xhtml.InlPres.type"/>
</xs:choice>
</xs:group>
<xs:group
name="xhtml.I18n.class">
<xs:sequence/>
</xs:group>
<xs:group
name="xhtml.Anchor.class">
<xs:sequence>
<xs:element
name="a"
type="xhtml.a.type"/>
</xs:sequence>
</xs:group>
<xs:group
name="xhtml.InlSpecial.class">
<xs:choice>
<xs:element
name="img"
type="xhtml.img.type"/>
<xs:element
name="object"
type="xhtml.object.type"/>
</xs:choice>
</xs:group>
</xs:group>
<xs:group name="xhtml.InlForm.class">
  <xs:choice>
    <xs:element name="input" type="xhtml.input.type"/>
    <xs:element name="select" type="xhtml.select.type"/>
    <xs:element name="textarea" type="xhtml.textarea.type"/>
    <xs:element name="label" type="xhtml.label.type"/>
    <xs:element name="button" type="xhtml.button.type"/>
  </xs:choice>
</xs:group>

<!-- Inline.class includes all inline elements, used as a component in mixes -->
<xs:group name="xhtml.Inline.class">
  <xs:choice>
    <xs:group ref="xhtml.InlStruct.class"/>
    <xs:group ref="xhtml.InlPhras.class"/>
    <xs:group ref="xhtml.InlPres.class"/>
    <xs:group ref="xhtml.Anchor.class"/>
    <xs:group ref="xhtml.InlSpecial.class"/>
    <xs:group ref="xhtml.InlForm.class"/>
    <xs:group ref="xhtml.Inline.extra"/>
  </xs:choice>
</xs:group>

<!-- InlinePre.mix Used as a component in pre model -->
<xs:group name="xhtml.InlinePre.mix">
  <xs:choice>
    <xs:group ref="xhtml.InlStruct.class"/>
    <xs:group ref="xhtml.InlPhras.class"/>
  </xs:choice>
</xs:group>
<xs:element name="tt"
type="xhtml.InlPres.type"/>
<xs:element name="i"
type="xhtml.InlPres.type"/>
<xs:element name="b"
type="xhtml.InlPres.type"/>
</xs:choice>
</xs:group>
<!--
InlNoAnchor.class includes all non-anchor inlines, used as a component in mixes
-->  
<xs:group name="xhtml.InlNoAnchor.class">
<xs:choice>
<xs:group ref="xhtml.InlStruct.class"/>
<xs:group ref="xhtml.InlPhras.class"/>
<xs:group ref="xhtml.InlPres.class"/>
<xs:group ref="xhtml.InlSpecial.class"/>
<xs:group ref="xhtml.InlForm.class"/>
<xs:group ref="xhtml.Inline.extra"/>
</xs:choice>
</xs:group>
<!--
InlNoAnchor.mix includes all non-anchor inlines
-->  
<xs:group name="xhtml.InlNoAnchor.mix">
<xs:choice>
<xs:group ref="xhtml.InlNoAnchor.class"/>
<xs:group ref="xhtml.Misc.class"/>
</xs:choice>
</xs:group>
<!--
Inline.mix includes all inline elements, including Misc.class
-->  
<xs:group name="xhtml.Inline.mix">
<xs:choice>
</xs:choice>
</xs:group>
In the HTML 4 DTD, heading and list elements were included in the block group. The Heading.class and List.class groups must now be included explicitly on element declarations where desired.

```xml
<xs:group name="xhtml.Heading.class">
  <xs:choice>
    <xs:element name="h1" type="xhtml.h1.type"/>
    <xs:element name="h2" type="xhtml.h2.type"/>
    <xs:element name="h3" type="xhtml.h3.type"/>
    <xs:element name="h4" type="xhtml.h4.type"/>
    <xs:element name="h5" type="xhtml.h5.type"/>
    <xs:element name="h6" type="xhtml.h6.type"/>
  </xs:choice>
</xs:group>
<xs:group name="xhtml.List.class">
  <xs:choice>
    <xs:element name="ul" type="xhtml.ul.type"/>
    <xs:element name="ol" type="xhtml.ol.type"/>
    <xs:element name="dl" type="xhtml.dl.type"/>
  </xs:choice>
</xs:group>
<xs:group name="xhtml.Table.class">
  <xs:choice>
    <xs:element name="table" type="xhtml.table.type"/>
  </xs:choice>
</xs:group>
```
</xs:choice>
</xs:group>
</xs:sequence/>
</xs:group/>

<!--
Block.class includes all block elements, used as an component in mixes
-->
<xs:group name="xhtml.Block.class">
<xs:choice>
<xs:group ref="xhtml.BlkStruct.class"/>
<xs:group ref="xhtml.BlkPhras.class"/>
<xs:group ref="xhtml.BlkPres.class"/>
<xs:group ref="xhtml.BlkSpecial.class"/>
<xs:group ref="xhtml.Block.extra"/>
</xs:choice>
</xs:group>

<!--
Block.mix includes all block elements plus %Misc.class;
-->
<xs:group name="xhtml.Block.mix">
<xs:choice>
<xs:group ref="xhtml.Heading.class"/>
<xs:group ref="xhtml.List.class"/>
<xs:group ref="xhtml.Block.class"/>
<xs:group ref="xhtml.Misc.class"/>
</xs:choice>
</xs:group>

<!--
All Content Elements
Flow.mix includes all text content, block and inline
Note that the "any" element included here allows us to add data from any other namespace, a necessity for compound document creation. Note however that it is not possible to add to any head level element without further modification. To add RDF metadata to the head of a document, modify the structure module.
-->
<xs:group name="xhtml.Flow.mix">
<xs:choice>
<xs:group ref="xhtml.Heading.class"/>
</xs:group>
</xs:group>
<xs:group ref="xhtml.List.class"/>
<xs:group ref="xhtml.Block.class"/>
<xs:group ref="xhtml.Inline.class"/>
<xs:group ref="xhtml.Misc.class"/>
</xs:choice>
</xs:group>
<xs:group name="xhtml.FlowNoTable.mix">
<xs:choice>
<xs:group ref="xhtml.Heading.class"/>
<xs:group ref="xhtml.List.class"/>
<xs:group ref="xhtml.BlkStruct.class"/>
<xs:group ref="xhtml.BlkPhras.class"/>
<xs:group ref="xhtml.Form.class"/>
<xs:group ref="xhtml.Inline.class"/>
<xs:group ref="xhtml.Misc.class"/>
</xs:choice>
</xs:group>

<!--
BlkNoForm.mix includes all non-form block elements,
plus Misc.class
-->
<xs:group name="xhtml.BlkNoForm.mix">
<xs:choice>
<xs:group ref="xhtml.Heading.class"/>
<xs:group ref="xhtml.List.class"/>
<xs:group ref="xhtml.BlkStruct.class"/>
<xs:group ref="xhtml.BlkPhras.class"/>
<xs:group ref="xhtml.BlkPres.class"/>
<xs:group ref="xhtml.Table.class"/>
<xs:group ref="xhtml.Block.extra"/>
<xs:group ref="xhtml.Misc.class"/>
</xs:choice>
</xs:group>
<xs:element name="html" type="xhtml.html.type"/>
</xs:schema>