XHTML Media Types - Second Edition

Serving the Most Appropriate Content to Multiple User Agents from a Single Document Source

W3C Working Group Note 16 January 2009

This version:
http://www.w3.org/TR/2009/NOTE-xhtml-media-types-20090116
Latest version:
http://www.w3.org/TR/xhtml-media-types
Previous version:
http://www.w3.org/TR/2002/NOTE-xhtml-media-types-20020801
Diff from previous version:
xhtml-media-types-diff.html
Editor:
Shane McCarron, Applied Testing and Technology, Inc.
First Edition Editor:
ISHIKAWA Masayasu
石川 雅康, W3C

Abstract

Many people want to use XHTML to author their web pages, but are confused about the best ways to deliver those pages in such a way that they will be processed correctly by various user agents. This Note contains suggestions about how to format XHTML to ensure it is maximally portable, and how to deliver XHTML to various user agents - even those that do not yet support XHTML natively. This document is intended to be used by document authors who want to use XHTML today, but want to be confident that their XHTML content is going to work in the greatest number of environments. The suggestions in this document are relevant to all XHTML Family Recommendations at the time of its publication.

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 Note made available by the World Wide Web Consortium (W3C) for your information. Publication as a Working Group Note does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

This document has been produced by the W3C XHTML 2 Working Group as part of the HTML Activity. The goals of the XHTML 2 Working Group are discussed in the XHTML 2 Working Group charter. The document represents working group consensus on the usage of Internet media types for various XHTML Family documents. However, this document is not intended to be a normative specification. Instead, it documents a set of recommendations to maximize the interoperability of XHTML documents with regard to Internet media types. This document *does not* address general issues on media types and namespaces.

Comments on this document may be sent to www-html-editor@w3.org (archive). Public discussion on this document may take place on the mailing list 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

1. Introduction

XHTML 1.0 [XHTML1] reformulated HTML 4 [HTML4] as an XML application, and XHTML Modularization [XHTMLM12N] provided a means to define XHTML-based

markup languages using XHTML modules, collectively called the "XHTML Family". However, due to historical reasons, a recommended way to serve such XHTML Family documents, in particular with regard to Internet media types, was somewhat unclear.

After the publication of [XHTML1], an RFC for XML media types was revised and published as RFC 3023 [RFC3023], and it introduced the '+xml' suffix convention for XML-based media types. The 'application/xhtml+xml' media type [RFC3236] was registered following that convention. There are several possible media types that can be used on XHTML Family documents. This Note addresses only the use of 'text/html' and 'application/xhtml+xml'.

This document summarizes the current best practice for using those various Internet media types for XHTML Family documents.

In general, 'application/xhtml+xml' should be used for XHTML Family documents, and the use of 'text/html' should be limited to HTML-compatible XHTML Family documents intended for delivery to user agents that do not explicitly state in their HTTP Accept header that they accept 'application/xhtml+xml' [HTTP]. The media types 'application/xml' and 'text/xml' may also be used, but whenever appropriate, 'application/xhtml+xml' or 'text/html' should be used rather than those generic XML media types.

Note that, because of the lack of explicit support for XHTML (and XML in general) in some user agents, only very careful construction of documents can ensure their portability (see Appendix A). If you do not require the advanced features of XHTML Family markup languages (e.g., XML DOM, XML Validation, extensibility via XHTML Modularization, semantic markup via XHTML+RDFa, Assistive Technology access via the XHTML Role and XHTML Access modules, etc.), you may want to consider using HTML 4.01 [HTML] in order to reduce the risk that content will not be portable to HTML user agents. Even in that case authors can help ensure their portability AND ease their eventual migration to the XHTML Family by ensuring their documents are valid [VALIDATOR] and by following the relevant guidelines in Appendix A.

2. Terms and Definitions

Note: While this document sometimes uses terms like "must" and "should", this document is not normative and those terms do not have the same meaning as when they are used in a normative W3C specification.

XHTML

The Extensible HyperText Markup Language. XHTML is *not* the name of a single, monolithic markup language, but the name of a family of document types which collectively form a family of related markup languages. The namespace URI for the XHTML Family is `http://www.w3.org/1999/xhtml`. Examples of document types that are members of this family include [XHTML1], and XHTML Host Language document types such as XHTML 1.1 [XHTML11] and XHTML Basic [XHTMLBasic]. Elements and attributes in those document types belong to the XHTML namespace (except those from the XML namespace, such as `xml:lang`), but an XHTML Family document type may also include elements and attributes from other namespaces, such as

MathML [MathML2].

3. Recommended Media Type Usage

This section summarizes which Internet media type should be used for XHTML Family documents. Note that while some suggestions are made in this section with regard to content delivery, this section is by no means a comprehensive discussion of content negotiation techniques.

That being said, a combination of these rules, in conjunction with a careful examination of the HTTP Accept header, can be useful in determining which media type to use when a document adheres to the guidelines in Appendix A. Specifically:

1. If the Accept header explicitly contains `application/xhtml+xml` (with either no "q" parameter or a positive "q" value) deliver the document using that media type.
2. If the Accept header explicitly contains `text/html` (with either no "q" parameter or a positive "q" value) deliver the document using that media type.
3. If the accept header contains "*/*" (a convention some user agents use to indicate that they will accept anything), deliver the document using `text/html`.

In other words, requestors that advertise they support XHTML family documents will receive the document in the XHTML media type, and all other requestors that (at least claim to) support HTML or "everything" will receive the document using the HTML media type. Dealing with user agents that satisfy none of these criteria is outside the scope of this document.

When an XHTML document does NOT adhere to the guidelines, it should only be delivered as media type `application/xhtml+xml`.

3.1. 'application/xhtml+xml'

The 'application/xhtml+xml' media type [RFC3236] is the **primary** media type for XHTML Family documents. 'application/xhtml+xml' should be used for serving XHTML documents to XHTML user agents (agents that *explicitly* indicate they support this media type). This media type must be used when writing documents using XHTML Family document types that add elements and attributes from foreign namespaces, such as XHTML+MathML [XHTML+MathML].

3.2. 'text/html'

The 'text/html' media type [RFC2854] is primarily for HTML, not for XHTML. In general, this media type is **NOT** suitable for XHTML except when the XHTML is conforms to the guidelines in Appendix A. In particular, 'text/html' is **NOT** suitable for XHTML Family document types that add elements and attributes from foreign namespaces, such as XHTML+MathML [XHTML+MathML].

XHTML documents served as 'text/html' will not be processed as XML [XML10], e.g., well-formedness errors may not be detected by user agents. Also be aware that HTML rules will be applied for DOM and style sheets (see guidelines 11 and 13).

Authors should also be careful about character encoding issues. See guideline 1 and guideline 9 for details.

Appendix A. Compatibility Guidelines

This appendix summarizes design guidelines for authors who wish their XHTML documents to render on **both** XHTML-aware and modern HTML user agents. The purpose of providing these guidelines is to supply a simple collection that, if followed, will give reasonable, predictable results in modern user agents. Document authors should treat these as best practices that were considered correct at the time this document was published. Like *all* of this document, this Appendix is *informative*. These guidelines in no way change the conformance requirements of any XHTML Family markup language.

Note that these guidelines were originally published in [XHTML1]. They have been moved into this document and updated because they are applicable beyond that recommendation, and also because they needed to be updated to reflect changes in modern user agents since that recommendation was first published.

For an example document that reflect the use of the guidelines from this section, see Appendix B.

A.1. Processing Instructions and the XML Declaration

DO NOT include XML processing instructions NOR the XML declaration.

Rationale: Some HTML user agents render XML processing instructions. Also, some user agents interpret the XML declaration to mean that the document is unrecognized XML rather than HTML. Such user agents may not render the document as expected. For compatibility with these types of HTML browsers, you should avoid using processing instructions and XML declarations.

Consequence: Remember, however, that when the XML declaration is not included in a document, AND the character encoding is not specified by a higher level protocol such as HTTP, the document can only use the default character encodings UTF-8 or UTF-16. See, however, guideline 9 below.

A.2. Elements that can never have content

If an element has an EMPTY content model DO use the minimized tag syntax permitted by XML (e.g., `<br />`). DO NOT use the alternative syntax (e.g., `<br></br>`) allowed by XML, since this may be unsupported by HTML user agents. Also, DO include a space

before the trailing `/` and `>`.

Empty elements in the XHTML family include: `area`, `base`, `basefont`, `br`, `col`, `hr`, `img`, `input`, `isindex`, `link`, `meta`, and `param`.

Rationale: HTML user agents ignore the `/>` at the end of a tag, but without it they may incorrectly parse the tag or its attributes. HTML user agents also may not recognize the alternate syntax permitted by XML.

A.3. *Elements that have no content*

If an element permits content (e.g., the `div` element) but an instance of that element has no content (e.g., an empty section), DO NOT use the "minimized" tag syntax (e.g., `<div />`).

Rationale: HTML user agents may give uncertain results when using the the minimized syntax permitted by XML when an element has no content.

A.4. Embedded Style Sheets and Scripts

DO use external style sheets if your style sheet uses `<` or `&` or `]]>` or `--`. DO NOT use an internal stylesheet if the style rules contain any of the above characters.

DO use external scripts if your script uses `<` or `&` or `]]>` or `--`. DO NOT embed a script in a document if it contains any of these characters.

Rationale: While XML provides the CDATA structure to embed data such as this, that method will not work correctly should the document be delivered as media type `text/html`. The best way to insulate your scripts and stylesheets is to make them external. Both XML and HTML have mechanisms that can protect this content when it is embedded in a document. If you must embed the content, the following patterns can be used:

Portably escaping embedded script contents:

```
<script>//<![CDATA[
...
//]]></script>
```

Portably escaping embedded style contents:

```
<style>/*<![CDATA[*/
...
/*]]>*/</style>
```

A.5. Line Breaks within Attribute Values

DO ensure that attribute values are on a single line and only use single whitespace characters. DO NOT use line breaks and multiple consecutive white space characters within attribute values.

Rationale: These are handled inconsistently by user agents - user agents are permitted to collapse multiple whitespace characters to a single white space character.

A.6. Deleted

This guideline was deleted because it is no longer relevant.

A.7. The `lang` and `xml:lang` Attributes

DO use both `lang` and `xml:lang` attributes when specifying the language of an element in markup languages that support the use of both.

Rationale: HTML 4 documents use the `lang` attribute to identify the language of an element. XML documents use the `xml:lang` attribute. CSS has a "lang" pseudo-selector that automatically uses the appropriate attribute depending on the media type. Therefore, specifying both attributes ensures that single CSS selectors will work in both modes.

A.8. Fragment Identifiers

DO use the `id` attribute to identify elements.

DO ensure that the values used for the `id` attribute are limited to the pattern `[A-Za-z][A-Za-z0-9:_.-]*`.

DO NOT use the `name` attribute to identify elements, even in languages that permit the use of `name` such as XHTML 1.0.

Rationale: In HTML 3.2 and earlier the `name` attribute on some elements could be used to define an anchor, but HTML 4 introduced the `id` attribute. In an XML dialect, only attributes with type `ID` are permitted to be used as anchors, and the `id` attribute is defined to be of type `ID`. Relying upon the `id` attribute as an anchor will work well in modern HTML and XHTML-aware user agents.

A.9. Character Encoding

DO encode your document in UTF-8 or UTF-16. When delivering the document from a server, DO set the character encoding for a document via the charset parameter of the

HTTP Content-Type header. When not delivering the document from a server, DO set the encoding via a "meta http-equiv" statement in the document (e.g., `<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />`). However, note that doing so will explicitly bind the document to an a single content type.

Rationale: Since these guidelines already recommend that documents NOT contain the XML declaration, setting the encoding via the HTTP header is the only reliable mechanism compatible with HTML and XML user agents. When that mechanism is not available, the only portable fallback is the "meta http-equiv" statement.

A.10. Boolean Attributes

DO use the full form for boolean attributes, as required by XML (e.g., `disabled="disabled"`). Such attributes include: `compact`, `nowrap`, `ismap`, `declare`, `noshade`, `checked`, `disabled`, `readonly`, `multiple`, `selected`, `noresize`, and `defer`.

Rationale: The compact form of these attributes is not well formed XML, and therefore invalid.

A.11. Document Object Model and XHTML

DO ensure examination of element and attribute names in scripts that use the DOM as defined in The Document Object Model level 1 Recommendation [DOM] are case-insensitive.

Rationale: This will ensure maximum portability of scripts, since the DOM methods will return element and attribute names in uppercase when served as `text/html` and in lowercase when served as `application/xhtml+xml`.

For example, in JavaScript you might do something like:

```
...
var name=node.name().toLowerCase;
if ( name == 'table' ) {
   ...
}
```

For more information on this, see also section 1.3 XHTML and the HTML DOM in [HTMLDOM2].

A.12. Using Ampersands

DO ensure that when content or attribute values contain the reserved character `&` it is used in its escaped form `&`.

Rationale: If ampersands are not encoded, the characters after them up to the next semi-colon can be interpreted as the name of a entity by the user agent. The document could also be considered not "well-formed" by an XML processor.

A.13. Cascading Style Sheets (CSS) and XHTML

DO use lower case element and attribute names in style sheets. DO create rules that include inferred elements (e.g., the `tbody` element in a table).

Rationale: These simple rules will help increase the portability of CSS rules regardless of the media type the document is processed as.

A.14. Referencing Style Elements when serving as XML

DO NOT use `xml-stylesheet` declarations to identify style sheets.

DO use the `style` or `link` elements to define stylesheets.

Rationale: Since XML processing instructions may be rendered by some HTML user agents, using the standard XML stylesheet declaration mechanism may not work well. However, since XHTML user agents are required to process `style` and `link` elements and interpret stylesheets referenced from those elements, documents constructed to use them will work as expected.

A.15. Formfeed Character in HTML vs. XML

DO NOT use the formfeed character (U+000C).

Rationale: This character is recognized as white space in HTML 4, but is NOT considered white space in all versions of XML.

A.16. The Named Character Reference '

DO use `&#39;` to specify an escaped apostrophe. DO NOT use `&apos;`.

Rationale: The entity `&apos;` is not defined in HTML 4.

A.17. The XML DTD Internal Subset

DO NOT use the XML DTD internal subset mechanism as part of the DOCTYPE declaration.

Rationale: The subset mechanism is not supported by non-XML user agents.

A.18. CDATA Sections

DO NOT use the XML CDATA mechanism excepting as described in guideline 4 above.

Rationale: This mechanism is not supported in non-XML user agents.

A.19. Explicit <tbody> Elements

DO use explicit `tbody` elements within tables.

Rationale: While the content model of the `table` element permits the `tbody` element to be skipped, in HTML 4 this element is *implicit*. HTML 4 user agents will silently add this element, thus potentially confusing scripts or style sheets.

A.20. `base` vs. `xml:base`

DO use the `base` element if you need to establish an alternate base URI for your document. DO NOT use the `xml:base` attribute.

Rationale: Most XHTML Family markup languages do not even support `xml:base`, relying instead upon the (X)HTML element `base`. HTML user agents do not support `xml:base` at all.

A.21. `document.write`

DO NOT use the JavaScript methods `document.write` or `document.writeln` to change the document. DO use DOM manipulation to achieve the same effect.

Rationale: Native XML user agents may not support this technique for modifying the content of the document.

A.22. Updating a document using `innerHTML`

DO NOT use the the JavaScript `innerHTML` method (supported by some user agents) to update a document dynamically.DO use standard DOM manipulation techniques instead. If you choose to use this method, ensure that it is available in the user agent and then ensure that any content inserted via the method is well-formed AND conforms to these guidelines so it is HTML 4 compatible.

Rationale: There are many other standard Document Object Model methods for updating

a document that will do so more portably than this non-standard method.

A.23. Scripts and missing `tbody` elements

DO ensure that if `tbody` elements are omitted from `table` elements, scripts that examine the document tree are capable of working both with and without the element.

Rationale: See guideline 19 above.

A.24. Styling the entire document

DO ensure that any CSS properties on the `html` element are also specified on the `body` element.

Rationale: Some HTML user agents only recognize CSS properties on the `body` element.

A.25. `noscript` Element

DO NOT use the `noscript` element.

Rationale: The contents of this element are treated differently depending on whether the document is evaluated as XML or HTML, as well as whether scripting is enabled in the user agent or not.

A.26. `iframe` Element

DO NOT embed content in the `iframe` element. DO use the `src` attribute to incorporate data via the `iframe` element.

Rationale: Content embedded in this element is treated differently depending on whether the document is evaluated as XML or HTML, as well as whether scripting is enabled in the user agent or not. The contents of the frame can be imported from an external source via the `src` attribute.

A.27. Creating elements in JavaScript

DO create new Document Object Model elements via the `createElementNS` method if it is available. If it is not, fallback to the `createElement` method.

Rationale: The createElementNS method will work as expected in both HTML and XML contexts, but it is not supported in all user agents.

Appendix B. An Example Document

The following is an example document that adopts the conventions described in Appendix A to ensure its portability among XHTML and HTML user agents. This example uses XHTML 1.0 Strict. It could also have used XHTML 1.1. In that case the top of the document would look like:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
                      "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
```

In this case, note that the `lang` attribute is not used on the `html` element. XHTML 1.1 does not support the `lang` attribute.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<!-- use lang and xml:lang if your language supports it - A.7 -->
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
  <title>Example Portable XHTML</title>
  <!-- Embed encoding when you can't set it from the server - A.9 -->
  <meta http-equiv="Content-Type"
        content="text/html; charset=utf-8" />

  <!-- use base, not xml:base - A.20 -->
  <base href="http://www.w3.org/MarkUp" />

  <!-- use an external stylesheet or script - A.4, A.14 -->
  <link href="style/style.css" rel="stylesheet" type="text/css" />

  <!-- if you must embed, escape bad stuff - A.4, A.14 -->
  <style type="text/css">/*<![CDATA[*/
      /* Use lower case names in stylesheets - A.13 */
      .main { padding-left: 10px; }
      /* put properties on whole document - A.24 */
      html {background-color: #e4e5e9; }
      body {background-color: #e4e5e9; }
  /*]]>*/</style>

  <!-- Need full form of boolean attributes like defer - A.10 -->
  <script defer="defer" type="text/javascript">//<![CDATA[
      var SOME_VALUE="<this is a test>";
  //]]></script>

</head>

<body>

<!-- use the id attribute to identify elements - A.8 -->
<div id="main">

    <!-- ensure there are never linebreaks in attribute values - A.5 -->
    <h1 title="This section shows how to write portable content">
        Main Section
    </h1>
    <!-- IMG can never have content so use shorthand form - A.2 -->
    <img src="http://www.w3.org/Icons/w3c_main" alt="W3C logo" />

    <!-- An element with no content cannot use shorthand - A.3 -->
    <p></p>

    <!-- Apostrophes must be escaped using numeric - A.16 -->
    <p title='rule for &#39;ampersands&#39;'>
```

```
            <!-- Ampersands must be escaped - A.12 -->
            Some material &amp; some other material

        </p>
        <table>
            <!-- use explicit tbody element - A.19 -->
            <tbody>
                <tr>
                    <td>Table Cell</td>
                </tr>
            </tbody>
        </table>

    </div>

    </body>
    </html>
```

Appendix C. References

[DOM]
"*Document Object Model (DOM) Level 1 Specification*", L. Wood *et al.*, 1 October 1998.
Available at: http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001

[HTMLDOM2]
"*Document Object Model (DOM) Level 2 HTML Specification*", J. Stenback *et al.*, 9 January 2003.
Available at: http://www.w3.org/TR/2003/REC-DOM-Level-2-HTML-20030109

[HTML4]
"HTML 4.01 Specification", W3C Recommendation, D. Raggett, A. Le Hors, I. Jacobs, eds., 24 December 1999. Available at: http://www.w3.org/TR/1999/REC-html401-19991224
The latest version of HTML 4.01 is available at: http://www.w3.org/TR/html401
The latest version of HTML 4 is available at: http://www.w3.org/TR/html4

[HTTP]
"Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, J. Gettys, R. Fielding, J. Mogul, H. Frystyk, L. Masinter, P. Leach and T. Berners-Lee, June 1999. Available at: http://www.rfc-editor.org/rfc/rfc2616.txt

[MathML2]
"Mathematical Markup Language (MathML) Version 2.0", W3C Recommendation, D. Carlisle, P. Ion, R. Miner, N. Poppelier, eds., 21 February 2001. Available at: http://www.w3.org/TR/2001/REC-MathML2-20010221
The latest version is available at: http://www.w3.org/TR/MathML2

[RFC2854]
"The 'text/html' Media Type", RFC 2854, D. Connolly, L. Masinter, June 2000. Available at: http://www.rfc-editor.org/rfc/rfc2854.txt

[RFC3023]
"XML Media Types", RFC3023, M. Murata, S. St.Laurent, D. Kohn, January 2001. Available at: http://www.rfc-editor.org/rfc/rfc3023.txt

[RFC3236]
"The 'application/xhtml+xml' Media Type", RFC 3236, M. Baker, P. Stark, January 2002. Available at: http://www.rfc-editor.org/rfc/rfc3236.txt

[VALIDATOR]
The W3C Markup Validation Service available at http://validator.w3.org.

[XHTML1]
"XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition): A Reformulation of HTML 4 in XML 1.0", W3C Recommendation, S. Pemberton et al., August 2002. Available at: http://www.w3.org/TR/2002/REC-xhtml1-20020801
The first edition is available at: http://www.w3.org/TR/2000/REC-xhtml1-20000126
The latest version is available at: http://www.w3.org/TR/xhtml1

[XHTML11]
"XHTML™ 1.1 - Module-based XHTML", W3C Recommendation, M. Altheim, S. McCarron, eds., 31 May 2001. Available at: http://www.w3.org/TR/2001/REC-xhtml11-20010531
The latest version is available at: http://www.w3.org/TR/xhtml11

[XHTMLBasic]
"XHTML™ Basic 1.1", W3C Recommendation, S. McCarron, M. Ishikawa eds., 29 July 2008. Available at: http://www.w3.org/TR/2008/REC-xhtml-basic-20080729
The latest version is available at: http://www.w3.org/TR/xhtml-basic

[XHTMLM12N]
"XHTML™ Modularization 1.1", W3C Recommendation, D. Austin et. al., eds., 8 October 2008. Available at: http://www.w3.org/TR/2008/REC-xhtml-modularization-20081008
The latest version is at: http://www.w3.org/TR/xhtml-modularization

[XHTML+MathML]
"XHTML plus Math 1.1 DTD", "A.2 MathML as a DTD Module", Mathematical Markup Language (MathML) Version 2.0. Available at: http://www.w3.org/TR/MathML2/dtd/xhtml-math11-f.dtd

[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

Appendix D. Changes from Previous Version

Restructured to be more focused on `text/html` and `application/xhtml+xml`

Migrated compatibility guidelines from XHTML 1.0 into this note so they can be updated and extended more easily.

Appendix E. Acknowledgements

This document is the work of many people over the course of many years. Unfortunately, it is impossible to properly acknowledge all of them. Suffice to say that they all have the thanks of the working group. A few contributors went above and beyond:

- Simon Pieters, Opera

At the time of this publication, the members of the XHTML 2 Working Group were:

- Roland Merrick IBM (XHTML 2 Working Group Co-Chair)
- Steven Pemberton, CWI (XHTML 2 Working Group Co-Chair)
- Mark Birbeck, webBackplane
- Susan Borgrink, Progeny Systems
- Christina Bottomley, Society for Technical Communication (STC)
- Alessio Cartocci, International Webmasters Association / HTML Writers Guild (IWA-HWG)
- Alexander Graf, University of Innsbruck
- Tina Holmboe, Greytower Technologies
- John Kugelman, Progeny Systems
- Luca Mascaro, International Webmasters Association / HTML Writers Guild (IWA-HWG)
- Shane McCarron, Aptest
- 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.