



HTML and XHTML Frequently Answered Questions

Editor: [Steven Pemberton](#), W3C/CWI

Version date: 21 July 2004

Other related FAQs:

- [Authoring Techniques for XHTML & HTML Internationalization 1.0](#)
- [W3C Web Internationalization FAQ](#)
- [Web Content Accessibility](#)
- [XForms 1.0 FAQ](#)
- [A list of all available W3C FAQs](#)

To comment on this document, or to send suggestions for questions, please email www-html-editor@w3.org, including the word FAQ in the subject.

Table of Contents

1. [Why is XHTML needed? Isn't HTML good enough?](#)
2. [What are the advantages of using XHTML rather than HTML?](#)
3. [Can I just put the XML declaration on top of existing HTML documents? Can I intermix HTML 4.01 and XHTML documents?](#)
4. [What is the easiest way to convert my HTML documents to XHTML?](#)
5. [Why are browsers so fussy about XML? They were more accepting with HTML.](#)
6. [Why should I care if my document is in correct HTML? It displays all right on my browser.](#)
7. [Where can I go to verify my document uses correct markup?](#)
8. [Why do you say "user agent" everywhere, instead of "browser"?](#)
9. [Why do I have to use these namespace things in XHTML?](#)
10. [Why is it allowed to send XHTML 1.0 documents as text/html?](#)
11. [Which browsers accept the media type application/xhtml+xml?](#)
12. [Does Microsoft Internet Explorer accept the media type application/xhtml+xml?](#)
13. [CSS has a lot of special rules that only apply to HTML. Do these also apply to XHTML?](#)
14. [Does document.write work in XHTML?](#)
15. [Why is it disallowed to send XHTML 1.1 documents as text/html?](#)
16. [Why was the target attribute removed from XHTML 1.1?](#)
17. [What is the use of XHTML Modularization?](#)
18. [Why is XHTML2 needed? Isn't XHTML 1 good enough?](#)
19. [Is being replaced by <object> in XHTML2?](#)
20. [Why doesn't XHTML2 use XLink?](#)
21. [Why isn't XHTML2 backwards compatible?](#)
22. [Why is xml:space set to 'preserve' on all elements of XHTML? I don't want to see extra space in my output.](#)

Why is XHTML needed? Isn't HTML good enough?

HTML is probably the most successful document markup language in the world. But when XML was introduced, a two-day workshop was organised to discuss whether a new version of HTML in XML was needed. The opinion at the workshop was a clear 'Yes': with an XML-based HTML other XML languages could include bits of XHTML, and XHTML documents could include bits of other markup languages. We could also take advantage of the redesign to clean up some of the more untidy parts of HTML, and add some new needed functionality, like better forms.

What are the advantages of using XHTML rather than HTML?

If your document is just pure XHTML 1.0 (not including other markup languages) then you will not yet notice much difference. However as more and more XML tools become available, such as XSLT for transforming documents, you will start noticing the advantages of using XHTML. XForms for instance will allow you to edit XHTML documents (or any other sort of XML document) in simple controllable ways. Semantic Web applications will be able to take advantage of XHTML documents.

If your document is more than XHTML 1.0, for instance including MathML, SMIL, or SVG, then the advantages are immediate: you can't do that sort of thing with HTML.

Can I just put the XML declaration on top of existing HTML documents? Can I intermix HTML 4.01 and XHTML documents?

No. HTML is not in XML format. You have to make the changes necessary to make the document proper XML before you can get it accepted as XML.

What is the easiest way to convert my HTML documents to XHTML?

[HTML Tidy](#) gives you the option to transform any HTML document into an XHTML one. [Amaya](#) is a browser/editor that will save HTML documents as XHTML.

Why are browsers so fussy about XML? They were more accepting with HTML.

This is deliberate. HTML browsers accept any input, correct or incorrect, and try to make something sensible of it. This error-correction makes browsers very hard to write, especially if all browsers are expected to do the same thing. It has also meant that huge numbers of HTML documents are incorrect, because since they display OK in the browser, the author isn't aware of the errors. This makes it incredibly difficult to write new web user agents since documents claiming to be HTML are often so poor.

Why should I care if my document is in correct HTML? It displays all right on my browser.

All browsers know how to deal with correct HTML. However, if it is incorrect, the browser has to repair the document, and since not all browsers repair documents in the same way, this introduces differences, so that your document may look and work differently on different browsers. Since there are hundreds of different browsers, and

more coming all the time (not only on PCs, but also on PDAs, mobile phones, televisions, printers, even refrigerators), it is impossible to test your document on every browser. If you use incorrect HTML and your document doesn't work on a particular browser, it is your fault; if you use correct HTML and it doesn't work, it is a bug in the browser.

Where can I go to verify my document uses correct markup?

W3C offers a service at <http://validator.w3.org/>. The [Amaya](#) browser/editor will also ensure that your markup is correct.

Why do you say "user agent" everywhere, instead of "browser"?

Although browsers are indeed important users of HTML and XHTML, there are other programs and systems that read those documents. Search engines for instance read documents, but are not browsers. By using the term "user agent" we are trying to remind people of the difference.

For example, when you do a Google search often you will see under some of the search results something like "This web page uses frames, but your browser doesn't support them." therefore surely frightening off some people from clicking on that link. The author of the website in question hasn't realised that there are more than just browsers, and that they ought to include better text in their `<noframes>` section, so that they don't appear so foolish when people search their site.

Why do I have to use these namespace things in XHTML?

In the early days of HTML different groups and companies added new elements and attributes to HTML at will. This threatened to cause a chaos of different non-interoperable versions of HTML. XML (the X stands for Extensible) allows anyone to use elements and elements from different languages, but for a browser or other user agent to know which element belongs to which language, you have to tell it. The namespace declarations do just that.

Why is it allowed to send XHTML 1.0 documents as text/html?

XHTML is an XML format; this means that strictly speaking it should be sent with an XML-related media type (`application/xhtml+xml`, `application/xml`, or `text/xml`). However XHTML 1.0 was carefully designed so that with care it would also work on legacy HTML user agents as well. If you follow some simple guidelines, you can get many XHTML 1.0 documents to work in legacy browsers. However, legacy browsers only understand the media type `text/html`, so you have to use that media type if you send XHTML 1.0 documents to them. But be well aware, sending XHTML documents to browsers as `text/html` means that those browsers see the documents as HTML documents, not XHTML documents.

Which browsers accept the media type application/xhtml+xml?

Browsers known to us include all Mozilla-based browsers, such as Mozilla, Netscape 5 and higher, Galeon and Firefox, as well as Opera, Amaya, Camino, Chimera, DocZilla, iCab, Safari, and all browsers on mobile phones that accept WAP2. In fact, any modern browser. Most accept XHTML documents as `application/xml` as well.

See the [XHTML Media-type test](#) for details.

Does Microsoft Internet Explorer accept the media type `application/xhtml+xml`?

No. However, there is a trick that allows you to serve XHTML1.0 documents to Internet Explorer as `application/xml`.

Include at the top of your document the line in bold here:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<?xml-stylesheet type="text/xsl" href="copy.xsl"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
```

where `copy.xsl` is a file that contains the following:

```
<stylesheet version="1.0"
  xmlns="http://www.w3.org/1999/XSL/Transform">
  <template match="/">
    <copy-of select="."/>
  </template>
</stylesheet>
```

Note that this file must be on the same site as the document referring to it.

Although you are serving the document as XML, and it gets parsed as XML, the browser thinks it has received `text/html`, and so your XHTML 1.0 document must follow many of the guidelines for serving to legacy browsers.

Your XHTML document will continue to work on browsers that accept XHTML 1.0 as `application/xml`.

CSS has a lot of special rules that only apply to HTML. Do these also apply to XHTML?

No. CSS rules that apply only to HTML, apply only to documents that are delivered as `text/html`.

Does `document.write` work in XHTML?

No. Because of the way XML is defined, it is not possible to do tricks like this, where markup is generated by scripting while the parser is still parsing the markup.

You can still achieve the same effects, but you have to do it by using the DOM to add and delete elements.

Why is it disallowed to send XHTML 1.1 documents as `text/html`?

XHTML 1.1 is pure XML, and only intended to be XML. It cannot reliably be sent to legacy browsers. Therefore XHTML 1.1 documents must be sent with an XML-related media type, such as `application/xhtml+xml`.

Why was the target attribute removed from XHTML 1.1?

It wasn't. XHTML 1.0 comes in three versions: strict, transitional, and frameset. All three of these were deliberately kept as close as possible to HTML 4.01 as XML would allow. XHTML 1.1 is an updated version of XHTML 1.0 *strict*, and no version of HTML strict has ever included the `target` attribute. The other two versions, transitional and frameset, were not updated, because there was nothing to update. If you want to use the `target` attribute, use XHTML 1.0 transitional.

What is the use of XHTML Modularization?

XHTML Modularization is not aimed at the regular users of XHTML, but at designers of XHTML-based languages. It had been observed that companies and groups had the tendency to design their own versions of HTML and XHTML that were often not interoperable at basic levels. XHTML Modularization splits XHTML into a number of modules that can be individually selected when defining a new language; in this way any XHTML-based language that uses tables is guaranteed to use the same definition of tables, and not some divergent version. Modularization also makes it clear where it is OK to add new elements, and where it is not.

Why is XHTML2 needed? Isn't XHTML 1 good enough?

HTML and XHTML have done good service, but there are many things that can be improved. Areas that have received particular attention include better structuring possibilities, removing features that are duplicated in XML, usability, accessibility, internationalization, device independence, better forms, and reducing the need for scripting.

Is being replaced by <object> in XHTML2?

No. `` is being replaced in XHTML2, but by something else (although you could use `<object>` if you wanted).

The design of `` has many problems in HTML:

- It has no fallback possibilities, so that if you use an image of type PNG for instance, and the browser can't handle that type, the only alternative is to use the `alt` text. This fact has hampered the adoption of PNG images, which in many ways are better than GIF and JPG, since people have continued to use the lowest-common denominator format, to ensure that everyone can see the images.
- The `alt` text cannot be marked up, so that if it gets used, you just get the plain text.
- It is possible to include a `longdesc` link to a description of the image, to help people who cannot see, but it is seldom implemented.

What XHTML2 does is say that all images are equivalent to some piece of content; it does this by allowing you to put a `src` attribute on any element at all. What this says is: if the image is available, and the browser can process it, use it, otherwise use the content of the element. For instance:

```
<p src="map.png">Exit from the station, turn left,
```

```
go straight on to <strong>High Street</strong>,
and turn right</p>
```

The advantage of this is that if the image is not available for some reason (such as network failure) or the browser can't render that sort of image, your document is still usable. If you want to supply more than one sort of image, you can do:

```
<p src="map.png"><span src="map.gif">Exit from station...</span></p>
```

although it is better to use content negotiation if your server supports it (and most do):

```
<p src="map">Exit from station...</p>
```

which would negotiate with the browser which sort of image it accepts, and give the browser its preferred sort. If there is no available image, then the content of the element would be used. This has an added advantage that you can later add other image types on your server and you don't have to change the page for it still to work.

Why doesn't XHTML2 use XLink?

XLink and XHTML had different requirements for linking that turned out not to be reconcilable.

Why isn't XHTML2 backwards compatible?

It is, but in a different way to how previous versions of HTML were backwards compatible.

Because earlier versions of HTML were special-purpose languages, it was necessary to ensure a level of backwards compatibility with new versions so that new documents would still be usable in older browsers. For instance, this is why the `<meta>` element has its content in an attribute rather than in the content of the element, since it would have shown up in older browsers.

However, thanks to XML and stylesheets, such strict element-wise backwards compatibility is no longer necessary, since an XML-based browser, of which at the time of writing means more than 95% of browsers in use, can process new markup languages without having to be updated. Much of XHTML 2 [works already in existing browsers](#), browsers that are not pre-programmed to accept XHTML2. Much works, but not all: when forms and tables were added to HTML, people had to wait for new version of browsers; similarly some parts of XHTML 2, such as XForms and XML Events, still require user agents that understand that functionality.

Why is `xml:space` set to 'preserve' on all elements of XHTML? I don't want to see extra space in my output.

The attribute `xml:space` is about *input*: that is to say, it controls if the spaces will be present in the DOM (i.e. in the internal version of the document inside the browser); it says nothing about what will appear on your screen. Output whitespace is controlled by the CSS property `'whitespace'`. Set it to `'pre'` and the spaces in the DOM will be preserved on output; set it to `'normal'` and the whitespace will be collapsed (CSS3 will have more properties to enable greater control).

This is the reason that all elements are set to `xml:space="preserve"` in XHTML2,

otherwise the CSS 'whitespace' property would have no effect, and you would have no control over visible whitespace. The default stylesheet will set 'whitespace' to 'normal' for all elements except `<pre>`, but you will be free to change them.

[Copyright](#) ©2004 [W3C](#)[®] ([MIT](#), [ERCIM](#), [Keio](#)), All Rights Reserved. W3C [liability](#), [trademark](#), [document use](#) and [software licensing](#) rules apply. Your interactions with this site are in accordance with our [public](#) and [Member](#) privacy statements.