

XHTML Workshop



28 January 2004

Helen Varley Sargan

Version 4.01 is the final version of HTML. XHTML is the bridging standard to make HTML accessible to XML applications (<http://www.w3.org/TR/xhtml1/>). XHTML 1.0 was the first of the XHTML family and comes in the same three 'flavours' as HTML, strict, transitional and frames. Use the transitional DTD for the least restrictive environment (see <http://www.w3.org/TR/xhtml1/>).

W3 cite the following as advantages for moving documents to XHTML:

- XHTML documents are XML conforming. As such, they are readily viewed, edited, and validated with standard XML tools.
- XHTML documents can be written to operate as well or better than they did before in existing HTML 4-conforming user agents as well as in new, XHTML 1.0 conforming user agents.
- XHTML documents can utilize applications (e.g. scripts and applets) that rely upon either the HTML Document Object Model or the XML Document Object Model [DOM].
- As the XHTML family evolves, documents conforming to XHTML 1.0 will be more likely to interoperate within and among various XHTML environments.

In addition, stripping out presentational mark-up and taking an XHTML/CSS approach allows your information to be much more accessible.

There have been several recommendations since XHTML 1.0 most of which are development steps and not to be used by general page authors:

- XHTML Basic has been designed for clients that don't support the full set of XHTML features, such as mobile phones, PDAs set-top boxes
- Modularized XHTML extends the approach set for XHTML basic (see <http://www.w3.org/MarkUp/modularization> for overview)
- XHTML 1.1, which is modularized reformulation of the XHTML 1.0 strict
- XHTML 2.0 is a working draft in development, released in May 2003 and is still under revision. This is a 'next generation' language and is not intended to be backward compatible with earlier versions.

None of these ought to be used by information providers wishing to communicate with a wide audience. The mark-up is too restrictive for the majority of browsers in use. XHTML 1.0 will give a useful stepping stone to more developed versions of XHTML once general browsers are able to cope.

XML is very restrictive compared with HTML, so XHTML imposes certain requirements.

Use stylesheets

Many formatting tags are no longer allowed and stylesheets provide a much better alternative formatting device, however you need to be aware of the following:

- CSS style sheets for XHTML should use lower case element and attribute names. Entity references as hex values should also be lowercase.
- In tables, the **tbody** element will be inferred by the parser of an HTML user agent, but not by the parser of an XML user agent. Therefore you should always explicitly add a **tbody** element if it is referred to in a CSS selector.
- Within the XHTML namespace, user agents are expected to recognize the "id" attribute as an attribute of type ID. Therefore, style sheets should be able to continue using the shorthand "#" selector syntax even if the user agent does not read the DTD.
- Within the XHTML namespace, user agents are expected to recognize the "class" attribute. Therefore, style sheets should be able to continue using the shorthand "." selector syntax.
- CSS defines different conformance rules for HTML and XML documents; be aware that the HTML rules apply to XHTML documents delivered as HTML and the XML rules apply to XHTML documents delivered as XML. (XHTML documents should be delivered as **text/html** for back compatibility)

XHTML differences

- **all mark-up needs to be lower case**
- **<head>, <body> and <title> elements are now mandatory**
- **all attribute values must be quoted**

this has always been good practice and often with newer browsers unquoted attributes in HTML will break web pages.

- **all tags must be closed**

unterminated tags such as `
` can be used in the form `
` (space then slash), which is a self-terminating tag - this form can be used for all unterminated tags, like `<img...>`, `<hr>` `<link>` `<meta>`. Although in XML terms the space is not required, older browsers need the space inserted to accommodate the trailing slash.

- **correct nesting of tags is required (no overlapping)**

this has always been good practice but html browsers have always been very tolerant of sloppy html. xhtml is not tolerant of, for instance:

```
<p><b>A good example of <i>hype</b></i></p>
```

which should of course be:

```
<p><b>A good example of <i>hype</i></b></p>
```

Also strictly governed is that inline tags must not contain block level tags, so

- `<a>` should not appear within other `<a>` tags nor `<form>` in other `<form>` tags
- `<pre>` tags should not contain ``, `<object>`, `<big>`, `<small>`, `<sub>` or `<sup>`

XHTML new requirements

- In order to be a valid XML document, XHTML pages must start with a `<!DOCTYPE>` element and contain the XHTML namespace declaration in the `<html>` element.

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"DTD/xhtml1-transitional.dtd" >
<html xmlns = "http://www.w3.org/1999/xhtml">
```

- **processing instructions (PI) ought to be added**
The PI is optionally the first item in any XML document. It looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
```

In this example, it does two things. It tells you (and any programs parsing the document) what version of XML the document is based on, and it declares the character encoding that the document is using. The PI is rendered in some HTML browsers, so you may want to leave it off if you can, and you can if the document only uses the default character encodings UTF-8 or UTF-16 (see next point).

- **language attribute and character encoding (further)**
Language attribute should be added in the `<html>` tag along with the namespace attribute:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
```

Character encoding should also be added as a meta statement in the `<head>` section (if it isn't added by your server via the `charset` parameter of the HTTP Content-Type header)

```
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1" />
```

- **empty attributes must have a value**
this applies to values such as the `html` tag `<hr noshade>`, which in `xhtml` should be written `<hr noshade="noshade" />` - older browsers (non-html4) might not take in the attribute at all and should fall back to the default for the tag.

- **certain elements now have additional mandatory attributes**
For instance, in XHTML every `` element must have an "alt" attribute. And every `<script>` element must have a "type" attribute.

- **using ampersands in attribute values**
If an attribute value contains an ampersand, it must be expressed as a character entity references (`&`), for instance:

```
<a href="http://oreillynet.com/cgi-bin/script.cgi?var=value&id=123">
```

should be

```
<a href="http://oreillynet.com/cgi-bin/script.cgi?var=value&amp;id=123">
```

- **scripts and style sheets are better handled as external files rather than inline**
this is because `xhtml` handles special characters (`<`, `>`, `[`, `]`, `-` and `&`) differently, which makes commenting out internal scripts and stylesheet information possible (in a CDATA section but not using standard comment tags) but only newer browsers can understand it. Put scripts and stylesheets as external files to overcome these problems.

- **fragment identifiers: name values are better handled as id values**
`` should be written as `` instead or as well (both can be used together without causing any difficulties). The use of `id` is built into the HTML 4.01 standard so is supported by newer and middle-aged browsers, but older browser will use **name**. To ensure backward compatibility strings matching the patterns `[A-Za-z][A-Za-z0-9:_.-]*` should be used

- **character entities, whitespace and end of lines in attribute values**

' was defined in XML1.0 but is not recognized in HTML4, so use ' instead. Avoid line breaks and multiple white space characters within attribute values. These are handled inconsistently by user agents.

Latin entity set, special characters and symbols allowed in XHTML 1.0 are listed in

http://www.w3.org/TR/xhtml1/dtds.html#a_dtd_Latin-1_characters,

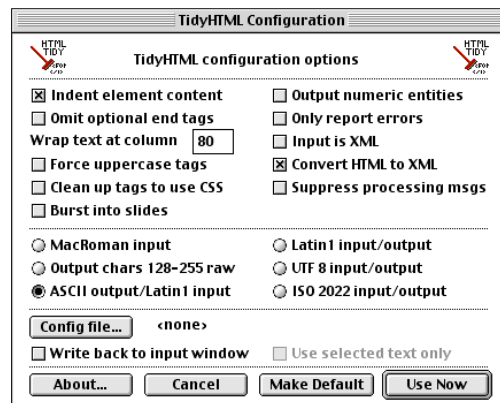
http://www.w3.org/TR/xhtml1/dtds.html#a_dtd_Special_characters and

http://www.w3.org/TR/xhtml1/dtds.html#a_dtd_Symbols, respectively.

A more extensive list may be found at <http://www.w3.org/TR/xhtml1/#guidelines>

Converting

HTML Tidy is a free tool that will sort out non-valid HTML code and convert files from HTML to XHTML (see <http://tidy.sourceforge.net/>). It is available as a tool both online and for a number of platforms, either as a standalone tool, integrated into a free html editor (several variations for Windows platforms) or part of (such as HomeSite) or to add into an existing html editor (such as BBEdit). The tool for BBEdit gives the following options:



There are a couple of web interfaces to Tidy as well, at

- The Dumb Terminal <http://www.thedumbterminal.co.uk/services/tidy.shtml>
- Jonathan Hedley's website <http://infohound.net/tidy/>

Don't assume because you have tidied a file it is valid - put it through a validator to check.

Validating

The W3 validator at <http://validator.w3.org/> will check html/xhtml files and give you a list of the problems. If it finds a big enough problem to start with (especially no DTD, no language declaration), it will have a fit and down tools, so you may have to follow through a sequence of correction steps. Some HTML editors have a validator or a link to a validator built in, but check exactly what it is before you depend on it.

Accessibility

Once you have a valid file, check the accessibility of it (see <http://bobby.watchfire.com/> or use the local PWF version of APrompt). If there are problems, you might be able to solve them by tweaking the settings in your HTML editor to remind you to add attributes to certain tags.