# XHTML

XHTML syntax is very similar to HTML syntax and almost all the valid HTML elements are valid in XHTML as well. But when you write an XHTML document, you need to pay a bit extra attention to make your HTML document compliant to XHTML.

The XHTML documents contains three parts, which are discussed below:

- **DOCTYPE:** It is used to declare a DTD
- **head:** The head section is used to declare the title and other attributes.
- **body:** The body tag contains the content of web pages. It consists many tags.

Here are the important points to remember while writing a new XHTML document or converting existing HTML document into XHTML document −

- Write a DOCTYPE declaration at the start of the XHTML document.
- Write all XHTML tags and attributes in lower case only.
- Close all XHTML tags properly.
- Nest all the tags properly/ All tags must be properly nested.
- Quote all the attribute values.
- Forbid Attribute minimization/Attributes cannot be shortened.
- Replace the name attribute with the id attribute.
- Deprecate the language attribute of the script tag.

Here is the detail explanation of the above XHTML rules −

## DOCTYPE Declaration

All XHTML documents must have a DOCTYPE declaration at the start. Creating a XHTML web page, it is necessary to include DTD (Document Type Definition) declaration. There are three types of DTD which are discussed below:

- Transitional DTD
- Strict DTD
- Frameset DTD

**Transitional DTD:** It is supported by the older browsers which does not have inbuilt cascading style sheets supports. There are several attributes enclosing the body tag which are not allowed in strict DTD.

**Syntax:**

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

**Strict DTD:** Strict DTD is used when XHTML page contains only markup language. Strict DTD is used together with cascading style sheets, because this attribute does not allow CSS property in body tag.
**Syntax:**

```
<!DOCTYPE html
```

```
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
```

**Frameset DTD:** The frameset DTD is used when XHTML page contains frames.

**Syntax:**

```
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
```

## Tags must be in lower case

XHTML is case-sensitive markup language. So, all the XHTML tags and attributes must be written in lower case.

```
<!-- Invalid in XHTML -->
<A Href="/xhtml/xhtml_tutorial.html">XHTML Tutorial</A>

<!-- Valid in XHTML -->
<a href="/xhtml/xhtml_tutorial.html">XHTML Tutorial</a>
```

## Closing Tags are mandatory

An XHTML must have an equivalent closing tag. Even empty elements should also have closing tags. Let's see an example:

```
<!-- Invalid in XHTML -->
<p>This paragraph is not written according to XHTML syntax.

<!-- Invalid in XHTML -->
<img src="/images/xhtml.gif" >

<!-- Valid in XHTML -->
<p>This paragraph is not written according to XHTML syntax.</p>

<!-- Valid in XHTML-->
<img src="/images/xhtml.gif" />
```

## Attribute Quotes

All the XHTML attribute's values must be quoted. Otherwise, your XHTML document is assumed as an invalid document.

See this example:

```
<!-- Invalid in XHTML -->
<img src="/images/xhtml.gif" width=250 height=50 />
```

```
<!-- Valid in XHTML -->
<img src="/images/xhtml.gif" width="250" height="50" />
```

## Attribute Minimization

XHTML doesn't allow you to minimize attributes. You have to explicitly state the attribute and its value.

See this example:

```
<!--Invalid in XHTML -->
<option selected>

<!-- valid in XHTML-->
<option selected="selected">
```

A list of minimized attributes in HTML and the way you need to write them in XHTML.

| HTML Style | XHTML Style |
|---|---|
| compact | compact="compact" |
| checked | checked="checked" |
| declare | declare="declare" |
| readonly | readonly="readonly" |
| disabled | disabled="disabled" |
| selected | selected="selected" |
| defer | defer="defer" |
| ismap | ismap="ismap" |
| nohref | nohref="nohref" |
| noshade | noshade="noshade" |
| nowrap | nowrap="nowrap" |
| multiple | multiple="multiple" |
| noresize | noresize="noresize" |

## The id Attribute

The id attribute is used to replace the name attribute. Instead of using name = "name", XHTML prefers to use id = "id".

See this example:

```
<!-- Invalid in XHTML -->
<img src="/images/xhtml.gif" name="xhtml_logo" />

<!-- Valid in XHTML -->
<img src="/images/xhtml.gif" id="xhtml_logo" />
```

## The language attribute

In XHTML, the language attribute of script tag is deprecated so you have to use type attribute instead of this.

See this example:

```
<!-- Invalid in XHTML -->
<script language="JavaScript" type="text/JavaScript">
  document.write("Hello XHTML!");
</script>

<!-- Valid in XHTML -->
<script type="text/JavaScript">
  document.write("Hello XHTML!");
</script>
```

# Nested Tags

XHTML tags must be nested properly. Otherwise your document is assumed as an incorrect XHTML document.

See this example:

```
<!-- Invalid in XHTML -->
<b><i> This text is bold and italic</b></i>

<!-- Valid in XHTML -->
<b><i> This text is bold and italic</i></b>
```

## Why use XHTML?

- XHTML documents are validated with standard XML tools.
- It is easily to maintain, convert, edit document in the long run.
- It is used to define the quality standard of web pages.
- XHTML is an official standard of the W3C, your website becomes more compatible and accurate with many browsers.

**Why Use XHTML?**

Developers who migrate their content to XHTML 1.0 get the following benefits −

- XHTML documents are XML conforming as they are readily viewed, edited, and validated with standard XML tools.
- XHTML documents can be written to operate better than they did before in existing browsers as well as in new browsers.
- XHTML documents can utilize applications such as scripts and applets that rely upon either the HTML Document Object Model or the XML Document Object Model.

- XHTML gives you a more consistent, well-structured format so that your webpages can be easily parsed and processed by present and future web browsers.
- You can easily maintain, edit, convert and format your document in the long run.
- Since XHTML is an official standard of the W3C, your website becomes more compatible with many browsers and it is rendered more accurately.
- XHTML combines strength of HTML and XML. Also, XHTML pages can be rendered by all XML enabled browsers.
- XHTML defines quality standard for your webpages and if you follow that, then your web pages are counted as quality web pages. The W3C certifies those pages with their quality stamp.

Web developers and web browser designers are constantly discovering new ways to express their ideas through new markup languages. In XML, it is relatively easy to introduce new elements or additional element attributes. The XHTML family is designed to accommodate these extensions through XHTML modules and techniques for developing new XHTML-conforming modules. These modules permit the combination of existing and new features at the time of developing content and designing new user agents.

## Benefits of XHTML:

- All XHTML tags must have closing tags and are nested correctly. This generates cleaner code.
- XHTML documents are lean which means they use less bandwidth. This reduces cost particularly if your web site has 1000s of pages.
- XHTML documents are well formated well–formed and can easily be transported to wireless devices, Braille readers and other specialized web environments.
- All new developments will be in XML (of which XHTML is an application).
- XHTML works in association with CSS to create web pages that can easily be updated.

## Difference Between HTML and XHTML:

| HTML | XHTML |
|---|---|
| ● HTML or HyperText Markup Language is the main markup language for creating web pages | ● XHTML (Extensible HyperText Markup Language) is a family of XML markup languages that mirror or extend versions of the widely used Hypertext Markup Language (HTML) |
| ● Flexible framework requiring lenient HTML specific parser | ● Restrictive subset of XML which needs to be parsed with standard XML parsers |
| ● Proposed by Tim Berners-Lee in 1987 | ● World Wide Web Consortium Recommendation in 2000. |
| ● Application of Standard Generalized Markup Language (SGML). | ● Application of XML |
| ● Extended from SGML. | ● Extended from XML, HTML |

## XHTML - Events

When users visit a website, they do things such as click on text, images and hyperlinks, hover-over things, etc. These are examples of what JavaScript calls events.

We can write our event handlers in JavaScript or VBScript and can specify these event handlers as a value of event tag attribute. The XHTML 1.0 has a similar set of events which is available in HTML 4.01 specification.

## The <body> and <frameset> Level Events

There are only two attributes which can be used to trigger any JavaScript or VBScript code, when any event occurs at document level.

| Attribute | Value | Description |
|---|---|---|
| onload | Script | Script runs when a XHTML document loads. |
| onunload | Script | Script runs when a XHTML document unloads. |

**Note** − Here, the script refers to any function or piece of code of VBScript or JavaScript.

## The <form> Level Events

There are following six attributes which can be used to trigger any JavaScript or VBScript code when any event occurs at form level.

| Attribute | Value | Description |
|---|---|---|
| onchange | Script | Script executes when the element changes. |
| onsubmit | Script | Script executes when the form is submitted. |
| onreset | Script | Script executes when the form is reset. |
| onselect | Script | Script executes when the element is selected. |
| onblur | Script | Script executes when the element loses focus. |
| onfocus | Script | Script runs when the element gets focus. |

## Keyboard Events

The following three events are generated by keyboard. These events are not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, and title elements.

| Attribute | Value | Description |
|---|---|---|

| | | |
|---|---|---|
| onkeydown | Script | Script executes on key press. |
| onkeypress | Script | Script executes on key press and release. |
| onkeyup | Script | Script executes key release. |

### Other Events

The following seven events are generated by mouse when it comes in contact with any HTML tag. These events are not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, and title elements.

| Attribute | Value | Description |
|---|---|---|
| onclick | Script | Script executes on a mouse click. |
| ondblclick | Script | Script executes on a mouse double-click. |
| onmousedown | Script | Script executes when mouse button is pressed. |
| onmousemove | Script | Script executes when mouse pointer moves. |
| onmouseout | Script | Script executes when mouse pointer moves out of an element. |
| onmouseover | Script | Script executes when mouse pointer moves over an element. |
| onmouseup | Script | Script executes when mouse button is released. |

## The differences

So how do the languages differ? In essence syntactically. While the syntax is very similar there are some important differences.

HTML is not case sensitive. XHTML is case sensitive, and all element and attribute names must be in lower case.

In HTML, some elements, such as paragraphs and list items can be implicitly closed. That means you may leave off the closing tag and you still have valid HTML. In XHTML, all elements must be explicitly closed.

Empty elements have a slightly different syntax. In HTML, an empty element looks like the start tag for any other element. In XHTML empty elements self close, by including a trailing slash before the closing angle bracket. Cunningly, this is in fact still valid HTML, although a space before the slash is recommended for compatibility with older browsers.

There are other small but important syntax differences as well. All attribute values must be quoted. In HTML, attributes without spaces in their values do not have to be quoted. Also, and this is a trick for many developers, attribute values in HTML allowed the use of unescaped special characters, such as the ampersand. In XHTML, such characters must be escaped in attribute values, just as they must be in the content of an element.