XHTML

Índice de contenido

Introducción

XML (lenguaje de marcas extensible) es también un metalenguaje (usado para crear otros lenguajes) y es también un sublenguaje de SGML, diseñado para ser más simple de procesar. En estos días, XML es ampliamente utilizado en diferentes formas para construir documentos y organizar información (por ejemplo, RSS (redifusión realmente simple), Atom, etc.) ya que provee una forma estándar de lograrlo que es más fácil de procesar que SGML.

En el año 2000, XHTML es recomendado por el World Wide Web Consortium (W3C) como la nueva versión estándar de HTML basada en XML en lugar de SGML. De esta forma, podemos considerar a XHTML como el resultado de mezclar HTML y XML. Hecho esto, todos los beneficios de XML son ahora heredados por HTML lo que lo hace más fácil de procesar, y por lo tanto estar disponible en más plataformas con capacidades de procesamiento reducidas (por ejemplo, PDAs (asistente digital personal) y teléfonos celulares).

Ventajas

Las principales ventajas del XHTML sobre otros formatos son:

- Compatibilidad parcial con navegadores antiguos: la información se visualiza, aunque sin formato. Cabe apuntar que el XHTML 1.0 fue diseñado expresamente para ser mostrado en navegadores que soportan HTML de base.

- Un mismo documento puede adoptar diseños radicalmente distintos en diferentes dispositivos, pudiendo incluso escogerse entre varios diseños para un mismo medio.

- Facilidad de edición directa del código y de mantenimiento.

- Formato abierto, compatible con los nuevos estándares que actualmente está desarrollando el W3C como recomendación para futuros agentes de usuario o navegadores.

- Los documentos escritos conforme a XHTML 1.0 pueden potencialmente presentar mejor rendimiento en las actuales herramientas web que aquellos escritos conforme a HTML..

Inconvenientes

- Algunos navegadores antiguos no son totalmente compatibles con los estándares, lo que hace que las páginas no siempre se muestren correctamente. Esto cada vez es menos problemático, al ir cayendo en desuso.

- Muchas herramientas de diseño web aún no producen código XHTML correcto.

Cosas a tener en cuenta para generar código XHTML

Etiquetas

- Las etiquetas no vacías deben ser cerradas siempre. No hay cierre opcional en XHTML.
  - Válido: `<p>Párrafo</p>`
  - Inválido: `<p>Párrafo`

- Las etiquetas vacías deben ser correctamente cerradas. Para lograr esto puedes usar un cierre normal o puedes cerrar la etiqueta poniendo un espacio seguido de una barra al final del tag de apertura.
  - Válido: `<img src="boton.jpg"></img><img src="boton.jpg" />`
  - Inválido: `<img src="boton.jpg">`

- Los nombres de etiquetas y atributos deben ser escritos en minúsculas para adaptarse a la sensibilidad a mayúsculas/minúsculas de XML (con la excepción del tag HTML !DOCTYPE).
  - Válido: `<a href="http://www.ua.es/es/index.html">Texto ancla</a>`
  - Inválido: `<A Href=" http://www.ua.es/es/index.html">Texto ancla</A>`

- Los elementos anidados deben obedecer correctamente su orden jerárquico. (el que se abre último, debe cerrarse primero)
  - Válido: `<span class="doble"><b>Ejecutar</b></span>`
  - Inválido: `<span class="doble"><b>Ejecutar</span></b>`

- Algunos elementos específicos no pueden ser declarados como contenido de otros elementos específicos.
  - El elemento "a" no debe contener otros elementos "a".
  - El elemento "pre" no debe contener otros elementos "img", "object", "big", "small", "sub" o "sup".
  - El elemento "button" no debe contener otros elementos "input", "select", "textarea", "label", "button", "form", "fieldset", "iframe" o "isindex".
  - El elemento "label" no debe contener otros elementos "label".
  - El elemento "form" no debe contener otros elementos "form".

- Todos los símbolos "&" deben ser escritos usando el nombre de entidad (&), aún en URLs.
  - Válido: `<a href="compraventa.php?id=1&sub=2">Compra & venta</a>`
  - Inválido: `<a href="compraventa.php?id=1&sub=2">Compra & venta</a>`

- Las referencias de entidad de caracteres son sensibles a cambios en mayúsculas/minúsculas de acuerdo a la regla de XML.
    - Válido: `&#xE1; - &aacute;` (para á)
    - Inválido: `&#XE1; - &aAcuTe;` (para á)

- El texto comentado será completamente ignorado por un procesador XML, lo que significa que comentar scripts o códigos de estilo para "ocultarlos" de los navegadores antiguos será igual a borrarlos. Si el script o código de estilo contiene un carácter "&" ó "<", éstos serán procesados por el procesador XML. Para evitar este inconveniente puedes elegir entre declararlos en archivos externos o utilizar el bloque CDATA.
    - Válido:
```
<style type="text/css">
<![CDATA[
 p { color: blue; }
]]>
</style>
```
    - Inválido:
```
<style type="text/css">
<!--
 p { color: blue; }
-->
</style>
```

- Los atributos desaprobados en HTML 4.01 no forman parte de XHTML (ver Apéndice I).
    - Inválido: `<font color="#0000FF">Blue text</font>`
    - Válido: `<span style="color: #0000FF;">Blue text</span>`

Atributos

- Los valores predefinidos de algunos atributos deben estar en minúsculas debido a la sensibilidad a mayúsculas/minúsculas de XML.
    - Válido: `<input type="submit" />`
    - Inválido: `<input type="SUBMIT" />`

- Los valores de los atributos deben ser adecuadamente encerrados entre comillas (simples o dobles). Las comillas no son opcionales en XHTML.
    - Válido: `<span id="id1" class='importante'>Texto</span>`
    - Inválido: `<span class=importante>Texto</span>`

- Los atributos booleanos no pueden ser abreviados (usando solo el nombre del atributo). Como valor debes especificar el nombre del atributo.
    - Válido: `<button id=”boton1” disabled=”disabled”>Ejecutar</button>`
    - Inválido: `<button id=”boton1” disabled>Ejecutar</button>`

En el siguiente ejemplo mostramos una definición de la etiqueta HTML button, compatible con código XHTML, con los atributos: "id", "disabled" (booleano) y "tabindex".

```
<button id="okbutton" disabled="disabled" tabindex="4">Ok</button>
```

- El atributo "alt" debe estar siempre presente en la etiqueta HTML img.
  - Válido: `<img src="pajaro.jpg" alt="Un pájaro volando"></img>`
  - Inválido: `<img src="pajaro.jpg"></img>`

- El atributo "name" ha sido formalmente desaprobado para los elementos a, applet, form, frame, iframe, img, y map, y puede ser excluido en futuras versiones.

Declaración del tipo de documento

Al escribir código XHTML hay dos cosas por considerar. La primera es que el atributo "lang" (para todos los tags que lo acepten) recibe una diferente denominación: "xml:lang". Esto no significa que los códigos de lenguaje sean diferentes, solo el nombre del atributo cambia. La segunda cosa por considerar, es que el atributo "xmlns" también debe ser definido para el tag html. Un documento normal compatible con código XHTML debería tener esta definición básica (XHTML 1.1):

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Esto es básico para XHTML 1.1, para XHTML 1.0 podemos tener 4 tipos de declaraciones según el tipo de documento.

- Estricto

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

- Transicional

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

- Con marcos

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

- Básico

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN" "http://www.w3.org/TR/xhtml-
basic/xhtml-basic10.dtd">
```

Declaración de nombre de espacio XML

La declaración de nombre de espacio XML es una simple URL (localizador uniforme de recursos) y puede ser definida como el valor del atributo "xmlns" para el tag html.

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

Reglas XHTML para DTDs estrictos

En adición a aquellas declaradas anteriormente, los documentos estrictos de XHTML (XHTML 1.0 Strict y XHTML 1.1) deben seguir estas reglas.

- El texto no debe ser definido directamente en el cuerpo del documento (tag HTML body). En lugar de ello, insértalo dentro de un párrafo, de un bloque div o de algún otro elemento.
  - Válido: `<body><p>Texto</p></body>`
  - Inválido: `<body>Texto</body>`

- Los elementos de bloque no pueden ser declarados como contenido de los elementos de línea.
  - Válido: `<div class="doble"><b>Ejecutar</b></div>`
  - Inválido: `<b><div class="doble">Ejecutar</div></b>`

XHTML 1.1.

Pero volviendo al tema, como decía, XML ya se ve muy cerca y tras la primera declaración digamos "XML real" de HTML son muchos los que empiezan a deshacerse de lenguajes de migración como XHTML1.0 Transicional para pasar a XML puro en HTML, o lo que es lo mismo, el paso de XHTML1.0 Strict a:
XHTML1.1, la primera reformulación de HTML en "XML de verdad"
Recomendación W3C de XHTML1.1 (31 mayo 2001) define un nuevo tipo de documento que está basado en un marco de módulos que están definidos en el documento de modularización de XHTML. Se busca que este nuevo tipo de documento sea la base para extender la familia XHTML y proveer consistencia, compatibilidad para aquellas opciones a eliminar (deprecated). Esta recomendación básicamente es una reformulación de XHTML 1.0 Strict incluyéndole el uso de módulos XHTML.

A. Cambios respecto a XHTML 1.0

- XHTML 1.1: XHTML Modular (10/4/2001)
- Se descompone la especificación en varios módulos
- XHTML Basic reúne un conjunto mínimo de módulos
  – Incluye images, forms, basic tables, y object support
- Un dispositivo puede soportar ciertos módulos: ej. un PDA, teléfono móvil, un set-top-box, dispositivo braille, sintentizador de voz, impresor, proyector,… pueden soportar XHTML 1.1
- Cada módulo puede extenderse y añadir nuevos
- Incorporación de: gráficos vectoriales (SVG), multimedia, MathML, comercio electrónico, …
  – Por ejemplo la funcionalidad de WML podría añadirse como módulo a XHTML 1.1

Diferencias con XHTML 1.0

1. En todos los elementos, el atributo lang ha sido eliminado en favor del atributo xml:lang
2. En los elementos a y map, el atributo name ha sido eliminado en favor del atributo id
3. La colección de elementos "ruby" ha sido añadida

XHTML 1.1 da de baja el soporte para:

Base	basefont	Center	font	Frame	frameset	Iframe
isindex	Menu	noframes	Object	s	strike	

Documentos Estrictamente Conformes

Un documento estrictamente conforme con XHTML 1.1 es un documento que requiere sólo las características como obligatorias en esta especificación. Por tanto, un documento debe seguir todos los criterios siguientes:

1. El elemento raíz del documento debe ser <html>.
2. El elemento raíz del documento debe designar el espacio de nombres XHTML usando el atributo xmlns. El espacio de nombres designado para XHTML es "http://www.w3.org/1999/xhtml".

3. Debe existir una declaración DOCTYPE en el documento previa al elemento raíz. Si está presente, el identificador público incluido en la declaración DOCTYPE debe hacer referencia a la DTD que se encuentra en el Apéndice C usando su Identificador Formal Público. El identificador system debe ser modificado apropiadamente.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Ejemplo de un documento XHTML 1.1.

```
<?xml version="1.0" encoding="UTF-8"?>
<!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" >
<head>
  <title>Virtual Library</title>
</head>
<body>
  <p>Moved to <a href="http://vlib.org/">vlib.org</a>.</p>
</body>
</html>
```

Notar que en este ejemplo, la declaración XML está incluida. Una declaración XML como la de arriba no está requerida en todos los documentos XML. Se anima fuertemente a autores de documentos XHTML al uso de declaraciones XML en todos sus documentos. En cualquier caso una declaración está requerida cuando la codificación de los caracteres del documento es distinta de las codificaciones por defecto UTF-8 o UTF-16.

El Tipo de Documento XHTML 1.1

El tipo de documento XHTML 1.1 es un tipo de documento completamente funcional con rica semántica. No es, en cualquier caso, tan variado en su funcionalidad como los tipos de documento XHTML 1.0 Transitional o Frameset. Dichos tipos de documento definían muchos componentes presentacionales que se manejan mejor mediante hojas de estilos u otros mecanismos similares. Más aún, partiendo de que el tipo de documento XHTML 1.1 está basado exclusivamente en las facilidades definidas en los módulos XHTML, no va a contener ninguna de las funcionalidades en desuso de XHTML 1.0 ni de HTML 4. Quitando estas excepciones, o quizá debido a ellas, el tipo de documento XHTML 1.1 es una base sólida para futuros tipos de documento que estén destinados a entornos de agentes de usuario variados.

El tipo de documento XHTML 1.1 está construido a partir de los siguientes módulos XHTML. Los elementos, atributos, y modelos mínimos de contenido asociados con dichos módulos son definidos en "Modularización de XHTML" (http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410). Los elementos son listados aquí con propósitos informativos, pero las definiciones de "Modularización de XHTML" deberían ser consideradas definitivas.

- Módulo de Estructura
  `body, head, html, title`
- Módulo de Texto *
  `abbr, acronym, address, blockquote, br, cite, code, dfn, div, em, h1, h2, h3, h4, h5, h6, kbd, p, pre, q, samp, span, strong, var`

- Módulo de Hipertexto
  `a`
- Módulo de Lista
  `dl, dt, dd, ol, ul, li`
- Módulo de Objetos
  `object, param`
- Módulo de Presentación
  `b, big, hr, i, small, sub, sup, tt`
- Módulo de Edición
  `del, ins`
- Módulo de Texto Bidireccional
  `bdo`
- Módulo de Formularios (básico 5 primeras)
  `form, input, label, select, option, textarea, button, fieldset, legend, optgroup`
- Módulo de Tablas
  `caption, table, td, th, tr, col, colgroup, tbody, tfoot, thead`
- Módulo de Imagen
  `img`
- Módulo de Mapa de Imagen del lado Cliente
  `area, map`
- Módulo de Mapa de Imagen del lado Servidor
  Atributo `ismap` de `img`
- Módulo de Eventos Intrínsecos
  Atributos de eventos
- Módulo de Metainformación
  `meta`
- Módulo de Scripting
  `noscript, script`
- Módulo de Hoja de Estilo
  `style`
- Módulo del Atributo Style (En desuso)
  `style` (atributo)
- Módulo de Link
  `link`
- Módulo de Base
  `base`

XHTML también utiliza el módulo Ruby Annotation:

- Ruby Annotation Module
  `ruby, rbc, rtc, rb, rt, rp`

XHTML 2.0

Recién hemos salido del cascaron de la guerra de los navegadores, los desarrolladores empiezan a entender las ventajas de un desarrollo utilizando las recomendaciones del W3C, y ahora es cuando se empieza a tomar en serio el XHTML como sustituto del antediluviano HTML 4.01.

Pero la verdad es que el W3C no descansa: el XHTML 1.1 fue publicado el 31 de Mayo del 2001 y desde entonces han estado lanzando continuamente borradores del futuro XHTML 2.0.

A los que no les apetezca la idea de tener un nuevo estándar que revisar no se alarmen: el camino a recorrer todavía es largo

El último borrador es de finales de Julio del 2006 y se empiezan a notar algunas diferencias abismales respecto al XHTML 1.0/1.1. Algunas de estas son:

Compatibilidad hacía atrás

En XHTML 2.0 se rompe la compatibilidad para atrás lo cual puede ser un gran handicap para su rápida expansión (la migración a XHTML 2.0 sera lenta, eso seguro).

Con esta ruptura el estándar sera mucho más pequeño, conciso y especifico, tendrá tendencia a ser más estructurado como XML, eliminará etiquetas de navegadores y se abstrae de ellos para servir mejor en dispositivos alternativos como PDA's.

Validación XML obligatoria

Obliga que las webs sean en formato estándar y elimina muchas barreras para el acceso a la información.

Obliga a tener unos conocimientos un poco más avanzados

Adiós Hx y DIV, hola SECTION y H

Los encabezados (<h1>, <h2>, <h3>, <h4>, <h5> y <h6>) son una limitación: ¿Qué pasa si necesitamos más niveles?…
Ahora mediante <section> se amplia en forma de árbol de contenidos hasta el infinito y por ello se sustituye por <h> (tal cual, sin número), por ejemplo:

```
<section>
   <h>Encabezado de primer nivel</h>

     <section>
       <h>Encabezado de segundo nivel</h>
     </section>
</section>
```

De la misma forma los son sustituidos por <section> aunque todavía no está claro si se podrán seguir usando o no.

Role

Cuando tengamos una parte de la web que no pertenece al contenido — por ejemplo: una lista de enlaces de navegación — podremos utilizar el atributo role:

```
<ul role="EnlacesNavegacion">
   <li>...</li>
</ul>
```

Aquí hay una diferencia y si lo he entendido bien es que se usara como si fuera un selector de CSS2, por lo que el CSS sería:

```
ul [role="EnlacesNavegacion"] {
   list-style: none;
}
```

Mas o menos role será es un sustituto de rel.

XFrames: Vuelven los marcos

Malas noticias, vuelven los frames, por lo que parece está nueva implementación soluciona el problema de las URLs inconsistentes

HREF extendido: Todo puede ser un enlace

El atributo href ya no forma parte únicamente de la etiqueta <a> por lo que puede usarse en cualquier etiqueta, las posibilidades pueden ser muchas por ejemplo las listas de enlaces, de:

```
<ul>
   <li><a href="link-1">texto del enlace</a></li>
   <li><a href="link-2">texto del enlace</a></li>
</ul>
```

Podemos pasar a:

```
<ul>
   <li href="link-1">texto del enlace</li>
   <li href="link-2">texto del enlace</li>
</ul>
```

XML includes

Se puede incluir contenido mediante ficheros XML sin necesidad de usar el include() de PHP, SSI , ASP o similares.

Un ejemplo:

```
<section id="navigation">
   <xi:include href="http://w3future.com/w3f/sections.xml" />
</section>
```

OBJECT

Object se utilizará para todo, sustituyendo las etiquetas style, img y applet

```
<object src="screen.css" srctype="text/css"></object>

<object src="foto.jpg"   srctype="image/jpeg"></object>
<object src="peli.avi"   srctype="video/x-msvideo"></object>
```

Atributo ALT

El atributo ALT de las imágenes desaparece y se puede especificar dentro del contenido ya que cualquier etiqueta puede tener el atributo “src=”, por ejemplo:

```
<p src="mifoto.jpg" type="image/jpeg">Está es mi foto.</p>
<object src="mifoto.jpg">Mi foto <em>salgo horrible</em></object>
```

Poder relacionar una imagen a un texto directamente nos ahorra el atributo ALT, el LONGDESC y añade semántica a la imagen, supongo que esto también ayudara a los motores de búsqueda a darle una mejor relevancia a las imágenes.

Cambios en HR y BR

El salto de línea <br /> parece que va a desaparecer en favor de <line /> y <hr> en favor de <separator />.

Apéndice I : Tags y atributos desaprobados en HTML

Los tags y atributos desaprobados son parte del estándar HTML (lenguaje de marcas hipertextual) cuyo uso ya no es recomendado. Esto sucede debido a que el estándar HTML es actualizado regularmente (por ejemplo, de la versión 3.0 a la 4.0) y algunos tags o características del lenguaje son agregados y otros removidos o desaprobados. Como consecuencia de esto, los autores de documentos HTML son provistos con más herramientas cuando un nuevo elemento o atributo es agregado, así como también advertidos de desechar aquellos que han sido reemplazados o se han vuelto obsoletos. La decisión de utilizar o no tags y/o atributos desaprobados es dejada a consideración de cada autor. Muchos navegadores ofrecen soporte para elementos desaprobados, pero en un futuro no muy lejano esto podría cambiar. La recomendación general es intentar utilizar otras formas de lograr sus efectos.

Los tags desaprobados en HTML 4.01 son fácilmente identificables. A continuación, mostramos una lista de los diez tags desaprobados:

- **applet** Reemplazado por el tag HTML object
- **dir** Reemplazado por el tag HTML ul
- **isindex** Reemplazado por el tag HTML input
- **menu** Reemplazado por el tag HTML ul

Tags desaprobados en favor de las hojas de estilo:

- **basefont**
- **center**
- **font**
- **s**
- **strike**
- **u**

Los atributos desaprobados varían de tag en tag. La mayoría de ellos son atributos de presentación y han sido desaprobados en favor de las hojas de estilo. Esto significa que su mismo efecto puede ser logrado usando el atributo "id", "style" o "class" en conjunto con hojas de estilo. En los ejemplos que a continuación se muestran, definimos dos trozos de código que lograrán el mismo efecto utilizando diferentes métodos.

Ejemplo desaprobado usando el tag HTML font:

```
<html>
<head>
<title>Ejemplo desaprobado para el tag HTML font</title>
</head>
<body>
...<font face="Arial,helvetica" color="#0000FF">trozo de texto en azul</font>...
</body>
</html>
```

Ejemplo utilizando el atributo "style":

```
<html>
<head>
<title>Ejemplo utilizando el atributo "style"</title>
</head>
<body>
...<span style="font-family: arial,helvetica; color: #0000FF">trozo de texto en
azul</span>...
</body>
</html>
```

Ejemplo utilizando clases con el tag HTML style:

```
<html>
<head>
<title>Ejemplo utilizando clases con el tag HTML style</title>
<style type="text/css">
span.bluetext {font-family: arial,helvetica; color: #0000FF}
</style>
</head>
<body>
...<span class="bluetext">trozo de texto en azul</span>...
</body>
</html>
```