## Resumen

Esta especificación define XHTML 1.0, una reformulación de HTML 4.0 como aplicación XML 1.0, y tres definiciones de tipo de documentos (DTD) correspondientes a las usadas en HTML 4.0. El significado de los elementos y sus atributos se encuentran en la Recomendación del W3C para HTML 4.0 y conforman la base para una futura extensión de XHTML. La compatibilidad con aplicaciones de usuario HTML existentes es posible siguiendo un pequeño conjunto de reglas.

## Estado de este documento

Esta sección describe el estado de este documento en el momento de su publicación. Otros documentos pueden dejarlo obsoleto. La última versión de esta serie de documentos se encuentra en la W3C.

Este documento ha sido revisado por los miembros del W3C y otras partes interesadas y ha sido avalado por el Director como una Recomendación del W3C. Es un documento estable y puede ser citado como material de referencia o referencia normativa en otro documento. El objetivo del W3C al hacer la Recomendación es llamar la atención sobre la especificación y promover su máxima difusión. Esto potencia la funcionalidad e interoperabilidad en la Web.

Este documento ha sido producido como parte de la [Actividad HTML del W3C](http://www.w3.org/MarkUp/). Las metas del[Grupo de Trabajo de HTML](http://www.w3.org/MarkUp/Group/)*(*[de acceso exclusivo a sus miembros](http://cgi.w3.org/MemberAccess/)*)* se discuten en la [carta del Grupo de Trabajo de HTML](http://www.w3.org/MarkUp/Group/HTMLcharter) *(*[de acceso exclusivo a sus miembros](http://cgi.w3.org/MemberAccess/)*)*.

Una lista actualizada de las Recomendaciones del W3C y otros documentos técnicos pueden encontrase en <http://www.w3.org/TR>.

La discusión pública sobre carácterísticas del HTML tiene lugar en la lista de correo [www-html@w3.org](mailto:www-html@w3.org) ([también disponible está disponible una recopilación](http://lists.w3.org/Archives/Public/www-html/)).

Por favor, notifiquen los errores que encuentren en este documento a [www-html-editor@w3.org](mailto:www-html-editor@w3.org).

Una lista de los errores de esta especificación puede encontrase en<http://www.w3.org/2000/01/REC-xhtml1-20000126-errata> (en inglés).

## Contenido

* 1. [¿Qué es XHTML?](#xhtml)
  + 1.1 [¿Qué es HTML 4.0?](#html4)
  + 1.2 [¿Qué es XML?](#xml)
  + 1.3 [¿Por qué es necesario XHTML?](#why)
* 2. [Definiciones](#defs)
  + 2.1 [Terminología](#terms)
  + 2.2 [Conceptos Generales](#general)
* 3. [Definición Normativa de XHTML 1.0](#normative)
  + 3.1 [Requisitos de Conformidad para Documentos](#docconf)
  + 3.2 [Requisitos de Conformidad para Aplicaciones de Usuario](#uaconf)
* 4. [Diferencias con HTML 4.0](#diffs)
* 5. [Compatibilidad](#issues)
  + 5.1 [Especificación del Tipo de Soporte de Internet](#media)
* 6. [Futuras Líneas de Actuación](#future)
  + 6.1 [Modularizar HTML](#mods)
  + 6.2 [Subconjuntos y Extensibilidad](#extensions)
  + 6.3 [Perfiles de Documento](#profiles)
* [Apéndice A. DTD](#dtds)
* [Apéndice B. Incompatibilidades entre elementos](#prohibitions)
* [Apéndice C. Directrices de compatibilidad con HTML](#guidelines)
* [Apéndice D. Agradecimientos](#acks)
* [Apéndice E. Referencias](#refs)

# 1. ¿Qué es XHTML?

XHTML es una familia de módulos y tipos de documentos que reproduce, engloba y extiende HTML 4.0 [[HTML]](#ref-html4). Los tipos de documentos de la familia XHTML están basados en XML, y diseñados fundamentalmente para trabajar en conjunto con aplicaciones de usuario basados en XML. Los detalles de esta familia y su evolución se discuten en más profundidad en la sección de [Futuras Directrices](#future).

XHTML 1.0 (esta especificación) es el primer tipo de documento de la familia XHTML. Es una reformulación de las tres definiciones de tipo de documento HTML 4.0 como aplicaciones de XML 1.0 [[XML]](#ref-xml). Su finalidad es ser usado como lenguaje de contenidos que es a la vez conforme a XML y, si se siguen algunas sencillas [directrices](#guidelines), funciona en aplicaciones de usuario conformes con HTML 4.0. Los desarrolladores que migren aplicaciones hacia XHTML 1.0 apreciarán las siguientes mejoras:

* Los documentos XHTML son conformes a XML. Como tales, son fácilmente visualizados, editados y validados con herramientas XML estándar.
* Los documentos XHTML pueden escribirse para que funcionen igual o mejor que lo hacían antes tanto en las aplicaciones de usuario conformes a HTML 4.0 como en los nuevas aplicaciones conformes a XHTML 1.0.
* Los documentos XHTML pueden usar aplicaciones (e.g. scripts y applets) que se basen ya sea en el Modelo del Objeto Documento de HTML o XML[[DOM]](#ref-dom).
* A medida que la familia XHTML evolucione, los documentos conformes a XHTML 1.0 estarán más preparados para interactuar dentro de y entre distintos entornos XHTML.

La familia XHTML es el siguiente paso en la evolución de Internet. Al migrar en este momento hacia XHTML, los desarrolladores de contenidos web entran en el mundo de XML con todos los beneficios que se esperan de él a la vez que se aseguran la compatibilidad con aplicaciones de usuario pasadas y futuras.

## 1.1 ¿Qué es HTML 4.0?

HTML 4.0 [[HTML]](#ref-html4) es una aplicación SGML (Lenguage de Etiquetado Generalizado Estándar) conforme al estándar internacional ISO 8879, y está ampliamente considerado como el lenguage de publicación estándar de la World Wide Web.

SGML es un lenguage para la descripción de lenguages de etiquetado, particularmente aquellos usados en el intercambio electrónico, manejo y publicación de documentos. HTML es un ejemplo de un lenguaje definido en SGML.

SGML es utilizado desde mitad de los 80 y ha permanecido bastante estable. Gran parte de su estabilidad se la debe al hecho de que el lenguaje es a la vez flexible y rico en posibilidades. Esta flexibilidad tiene sin embargo su coste, el nivel de complejidad que ha inhibido su uso en diversos ámbitos como la World Wide Web.

HTML, tal y como fue concebido, era un lenguaje para el intercambio de documentos científicos y técnicos adaptado para su uso por no especialistas en tratamiento de documentos. HTML resolvió el problema de la complejidad de SGML sirviéndose de un reducido conjunto de etiquetas estructurales y semánticas apropiadas para la realización de documentos relativamente simples. Además de simplificar la estructura de los documentos, HTML soportaba el hipertexto. Las posibilidades de usar elementos multimedia fueron añadidas con posterioridad.

En un corto período de tiempo, HTML se hizo muy popular y rápidamente superó los propósitos para los que había sido creado. Desde los albores de HTML , ha habido una constante invención de nuevos elementos para ser usados dentro de HTML (como estándar) y para adaptar HTML a mercados verticales, altamente especializados. Esta plétora de nuevos elementos ha llevado a problemas de compatibilidad de los documentos en las distintas plataformas.

Dada la creciente heterogeneidad de programas y plataformas , está claro que la idoneidad del HTML 4.0 'clásico' para ser usado en dichas plataformas es más que limitado.

## 1.2 ¿Qué es XML?

XML™ son las siglas de Lenguaje de Etiquetado Extensible , fórmandose la palabra como acrónimo de la expresión inglesa 'eXtensible Markup Language' [[XML]](#ref-xml).

XML fue concebido como un medio para recobrar la potencia y flexibilidad de SGML sin que adquiriese su gran complejidad. A pesar de ser una forma restringida de SGML, XML conserva casi toda la potencia y riqueza de las carácterísticas de SGML.

Aún manteniendo estas carácterísticas, XML elimina las más complejas de SGML que hacían la creación y diseño de los programas apropiados dificil y costosa.

## 1.3 ¿Por qué es necesario XHTML?

Los beneficios de migrar hacia XHTML 1.0 se han descrito más arriba. Algunos de esos beneficios generales son:

* Los desarrolladores de aplicaciones de usuario y documentos descubren constantemente nuevas formas de expresar sus ideas usando nuevas etiquetas. En XML, es relativamente fácil añadir nuevos elementos así como atributos adicionales a dichos elementos o a los ya existentes. La familia XHTML está concebida para acomodar estas extensiones a través de módulos XHTML y técnicas para desarrollar nuevos módulos conformes a XHTML (tal y cómo se describirá en la futura especificación sobre Modularización XHTML). Estos módulos permitirán la combinación de las carácterísticas existentes con las nuevas al crear contenidos para la web así como al desarrollar nuevas aplicaciones de usuario.
* Constantemente se desarrollan nuevas formas de acceder a Internet. Algunas estimaciones indican que en el año 2002, un 75% de las peticiones de documentos que se visualicen en Internet se realizarán desde esas plataformas alternativas. La familia XHTML está concebida teniendo en mente la interoperabilidad con aplicaciones de usuario generales. A través de un nuevo mecanismo de especificación de documentos y aplicaciones de usuarios , los servidores, proxies, y aplicaciones de usuario finales podrán realizar una mejor transformación del contenido. Como objetivo final, será posible desarrollar contenido conforme a XHTML que sea utilizable por cualquier aplicación de usuario conforme a XHTML.

# 2. Definiciones

## 2.1 Terminología

Los siguientes términos se utilizan en esta especificación. Estos términos extienden las definiciones de [[RFC2119]](#ref-rfc2119) basándose en definiciones similares que aparecen en ISO/IEC 9945-1:1990 [[POSIX.1]](#ref-posix):

**Definido por la aplicación**

Un valor o comportamiento se considera definido por la aplicación cuando esta última es la encargada de definir [y documentar] los requisitos correspondientes para la correcta construcción del documento.

**Puede**

Con respecto a las aplicaciones, la palabra "puede" debe entenderse como introductoria a una carácterística opcional que no es obligatoria para hacer conforme el documento a esta especificación pero que puede indicarse. Con respecto a los [Requisitos de Conformidad para Documentos](#docconf), la palabra "puede" significa que la carácterística opcional que introduce no debe utilizarse. El término "opcional" tiene la misma definición que "puede".

**Debe**

En esta especificación la palabra "debe" denota un requisito obligatorio ya sea de la aplicación o de los Documentos XHTML Estrictamente Conformes con la especificación, dependiendo del contexto. El término "deberá" tiene el mismo sentido que "debe".

**Reservado**

Denota un valor o comportamiento que no está especificado y cuyo uso tampoco está permitido en Documentos Conformes ni debe ser aceptado por Aplicaciones de Ususario Conformes.

**Debería**

Con respecto a las aplicaciones, la palabra "debería" ha de intrepretarse como una recomendación para la aplicación, pero no como un requisito. Con respecto a los documentos, este término ha de enterderse como una práctica de programación deseable en general y un requisito para Documentos XHTML Estrictamente Conformes en particular.

**Admitido**

Algunos recursos recogidos en esta especificación son opcionales. Si uno de ellos es admitido por alguna aplicación, se comporta como se indica en esta especificación.

**No determinado**

Cuando un valor o comportamiento no quedan determinados, la especificación no define requisitos de portabilidad para un recurso en una aplicación ni siquiera cuando deba analizar un documento que use dicho recurso. Un documento tal que requiera un comportamiento específico en dicha situación en vez de tolerar cualquier comportamiento al usar el recurso, no será un Documento XHTML Estrictamente Conforme.

## 2.2 Conceptos Generales

**Atributo**

Un atributo es un parámetro de un elemento declarado en la DTD. El tipo de un atributo y su rango de valores, incluyendo la posibilidad de un valor por defecto, se definen en la DTD.

**DTD**

Una DTD, o definición del tipo de documento, es una colección de declaraciones XML que, como colección, define la estructura reglamentaria, los *elementos* y *atributos* que están disponibles para su uso en documentos que cumplan con la DTD.

**Documento**

Un documento es una cadena de datos que, tras ser combinado con cualquier otra cadena a la que referencie, queda estructurado de tal manera que porta información contenida en*elementos* que se organizan tal y como está especificado en la correspondiente *DTD*. Para más información, ver los [Requisitos de Conformidad para Documentos](#docconf).

**Elemento**

Un elemento es una unidad estructural de un documento que ha sido declarada en la *DTD*. El modelo de contenidos del elemento está definido en la *DTD*, y la explicación adicional puede especificarse en la descripción comentada del elemento.

**Recursos**

Los componentes disponibles incluyen *elementos*, *atributos* y las explicaciones asociadas a dichos *elementos* y *atributos*. Una aplicación que admita dichos componentes debe facilitar los recursos necesarios para procesarlos.

**Aplicación**

Una aplicación es una sistema que posee una colección de *recursos* y servicios que admite esta especificación. Ver el apartado de [Conformidad de Aplicaciones de Usuario](#uaconf) para más información.

**Análisis**

El análisis es el proceso por el cual un *documento* es leído y la información en él contenida se traduce en el contexto de *elementos* en que esta información está estructurada.

**Presentación**

La presentación es el proceso por el cual la información contenida en un *documento* se muestra al usuario. Esto se lleva a cabo de la forma más apropiada al entorno que utilice el usuario (e.g. de forma auditiva, visual, impresa).

**Aplicación de Usuario**

Una aplicación de usuario es una *aplicación* que lee y procesa documentos XHTML. Ver[Conformidad de Aplicaiones de Usuario](#uaconf) para más información.

**Convalidación**

La convalidación es un proceso por el cual los *documentos* son contrastados con la *DTD*asociada, asegurándose de que la estructura, el uso de *elementos* y el uso de *atributos* son consistentes con las definiciones de la *DTD*.

**Gramaticalidad (Documento "bien formado")**

Un *documento* se dice "bien formado" o "gramaticalmente correcto" cuando está estructurado de acuerdo a las reglas definidas en la [Sección 2.1](http://www.w3.org/TR/REC-xml#sec-well-formed) de la Recomendación de XML 1.0 [[XML]](#ref-xml). Básicamente, en dicha definición implica que los elementos, delimitados por sus etiquetas de inicio y fin, estén convenientemente anidados.

# 3. Definición Normativa de XHTML 1.0

## 3.1 Requisitos de Conformidad para Documentos

Esta versión de XHTML suministra una definición de documentos XHTML estrictamente conformes que se restringe a las etiquetas y atributos del espacio nominal de XHTML. Ver la[Sección 3.1.2](#well-formed) para información concerniente al uso de XHTML con otros espacios nominales, por ejemplo la inclusión de metadatos expresados en RDF dentro de documentos XHTML.

### 3.1.1 Documentos Estrictamente Conformes

Un documento XHTML estrictamente conforme es un documento que para ser procesado requiere tan sólo los recursos descritos como obligatorios en esta especificación. Tales documentos deben ajustarse a los siguientes puntos:

1. Deben poder validarse con alguna de las tres DTD que se encuentran en el [ApéndiceA](#dtds).
2. El elemento raíz del documento debe ser <html>.
3. El elemento raíz del documento debe indicar el espacio nominal XHTML usando el atributoxmlns [[XMLNAMES]](#ref-xmlns). El espacio nominal para XHTML es http://www.w3.org/1999/xhtml
4. Debe haber una declaración DOCTYPE en el documento antes del elemento raíz. El identificador público incluido en la declaración DOCTYPE debe hacer referencia a alguna de las tres DTD que se hallan en el [Apéndice A](#dtds) usando el Identificador Formal Público correspondiente. El identificador del sistema puede ser modificado apropiadamente para reflejar convenciones de rango local.

<!DOCTYPE html

PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!DOCTYPE html

PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<!DOCTYPE html

PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

He aquí un ejemplo de un pequeño documento XHTML.

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

<!DOCTYPE html

PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

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

<head>

<title>Biblioteca Virtual</title>

</head>

<body>

<p>Trasladada a <a href="http://vlib.org/">vlib.org</a>.</p>

</body>

</html>

Nótese que en este ejemplo, se incluye la declaración XML. Una declaración XML como la que se hace arriba no es necesaria en todos los documentos XML. Aún así, se recomienda encarecidamente a los autores de documentos XHTML que incluyan declaraciones XML en todos sus documentos. Tal declaración es necesaria cuando la codificación de los caracteres que se usa en el documento no es UTF-8 o UTF-16, los tipos usados por defecto en este tipo de documentos.

### 3.1.2 Usando XHTML con otros espacios nominales

El espacio nominal XHTML 1.0 puede usarse conjuntamente a otros espacios nominales XML como se indica en [[XMLNAMES]](#ref-xmlns), aunque los documentos así producidos no serán documentos XHTML 1.0 estrictamente conformes. Futuros trabajos del W3C darán directrices que especifiquen la conformidad de documentos que usen varios espacios nominales.

El siguiente ejemplo muestra cómo XHTML 1.0 podría usarse en conjunción con la Recomendación MathML:

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

<head>

<title>Un ejemplo matemático</title>

</head>

<body>

<p>Lo que viene a continuación es etiquetado MathML:</p>

<math xmlns="http://www.w3.org/1998/Math/MathML">

<apply> <log/>

<logbase>

<cn> 3 </cn>

</logbase>

<ci> x </ci>

</apply>

</math>

</body>

</html>

El siguiente ejemplo muestra cómo el etiquetado XHTML 1.0 podría usarse en otro espacio nominal XML:

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

<!-- inicialmente, el espacio nominal por defecto es "books" -->

<book xmlns='urn:loc.gov:books'

xmlns:isbn='urn:ISBN:0-395-36341-6' xml:lang="en" lang="en">

<title>Cheaper by the Dozen</title>

<isbn:number>1568491379</isbn:number>

<notes>

<!-- hacemos HTML el espacio nominal por efecto para un comentario

hipertextual -->

<p xmlns='http://www.w3.org/1999/xhtml' xml:lang="sp" lang="sp">

También está disponible <a href="http://www.w3.org/">en la red</a>.

</p>

</notes>

</book>

## 3.2 Requisitos de Conformidad para Aplicaciones de Usuario

Una aplicación de usuario debe cumplir todos y cada uno de los siguientes criterios de conformidad:

1. Para ser consistente con la Recomendación XML 1.0 [[XML]](#ref-xml), la aplicación de usuario debe analizar y evaluar si un documento XHTML es "gramaticalmente correcto". Si el aplicación de usuario se dice convalidante debe contrastar los documentos con las DTD de acuerdo con [[XML]](#ref-xml).
2. Cuando una aplicación de usuario dice soportar [recursos](#facilities) definidos en esta especificación o requeridos por esta especificación a través de referencia normativa debe hacerlo de manera consistente con la definición del recurso.
3. Cuando una aplicación de usuario procesa un documento XHTML como XML genérico, tan sólo debería reconocer atributos del tipo ID (e.g. el atributo id de la mayoria de los elementos XHTML) como identificadores de fragmentos.
4. Si una aplicación de usuario encuentra un elemento que no reconoce, debe presentar el contenido de dicho elemento.
5. Si una aplicación de usuario encuentra un atributo que no reconoce, debe ignorar completamente la directriz que marque el atributo (i.e., el atributo y su valor).
6. Si una aplicación de usuario encuentra un valor de un atributo que no reconoce, debe usar en su lugar el valor por defecto del atributo.
7. Si encuentra una referencia a una entidad (distinta a las entidades predefinidas) para la que la aplicación de usuario no ha procesado ninguna declaración (lo que podría suceder si la declaración se encuentra en un subconjunto externo al que la aplicación de usuario no ha accedido), la entidad debe presentarse como los caracteres (comenzando con & y terminando con punto y coma) que componen la referencia a la entidad.
8. Cuando se presente el contenido, las aplicaciones de usuario que encuentren caracteres o referencias a entidades de tipo carácter que reconozcan pero que no sean capaces de mostrar, deberían mostrar el documento de tal manera que el usuario aprecie claramente que no ha sido posible una presentación correcta.
9. Los siguientes caracteres se definen en [XML] como caracteres de espacios en blanco:
   * Espacio (&#x0020;)
   * Tabulación (&#x0009;)
   * Retorno de carro (&#x000D;)
   * Avance de línea (&#x000A;)

El procesador XML normaliza varios sistemas de códigos de fin de línea en un único carácter de avance de línea que se pasa a la aplicación. La aplicación de usuario XHTML debe, además, tratar los siguientes caracteres como espacios en blanco:

* + Salto de página (&#x000C;)
  + Espacio de anchura nula (&#x200B;)

En elementos donde el atributo 'xml:space' tenga el valor 'preserve', la aplicación de usuario debe conservar intactos todos los espacios en blanco (con excepción de los caracteres de espacio en blanco de encabezamiento y terminación, que deberían ser suprimidos). En otros casos un espacio en blanco se manipula de acuerdo con las siguientes reglas:

* + Todos los espacios en blanco que rodean a un elemento en bloque deberían ser suprimidos.
  + Los comentarios se suprimen por completo y no afectan a la manipulación de espacios en blanco. Un carácter de espacio en blanco a ambos lados de un comentario se trata como dos espacios en blanco.
  + Los espacios en blanco de encabezamiento y terminación dentro de un elemento en bloque deben suprimirse.
  + Los caracteres de avance de línea dentro de un elemento en bloque deben convertirse en un espacio (excepto cuando el atributo 'xml:space' tenga el valor 'preserve').
  + Una secuencia de espacios en blanco debe reducirse a un solo carácter de espacio en blanco (excepto cuando el atributo 'xml:space' tenga el valor 'preserve').
  + Con respecto a la presentación, la aplicación de usuario debería presentar el contenido de manera adecuada al idioma en el que el contenido está escrito. En idiomas cuya base de escritura sea de caracteres latinos, el carácter ASCII de espacio se usa normalmente bien como límite gramatical entre palabras bien como espacio tipográfico en blanco; en idiomas cuya base esté relacionada con Nagari (e.g., Sanskrit, Thai, etc.), los límites gramaticales pueden codificarse con el carácter de 'espacio' ZW, pero no serán normalmente representados por caracteres tipográficos de espacio en blanco al ser mostrados; los idiomas que usen una base de escritura arábica pueden codificar el carácter tipográfico espacio en blanco usando un carácter de espacio en blanco, pero también pueden usar el carácter de 'espacio' ZW para delimitar límites gramaticales 'internos' (lo que parecen palabras en árabe para un lector inglés frecuentemente engloba varias palabras, e.g. 'kitAbuhum' = 'kitAbu-hum' = 'libro de ellos' == su libro); e idiomas provenientes del chino normalmente no codifican tales delimitadores ni usan el carácter tipográfico de espacio en blanco de este modo.

El espacio en blanco en los valores de atributos se procesa de acuerdo con [[XML]](#ref-xml).

# 4. Diferencias con HTML 4.0

Debido al hecho de que XHTML es una aplicación XML, ciertas prácticas que eran perfectamente válidas en HTML 4.0 [[HTML]](#ref-html4),basado en SGML, deben cambiar.

## 4.1 Los documentos deben ser "gramaticalmente correctos"

La [gramaticalidad](#wellformed) de los documentos es un nuevo concepto introducido por [[XML]](#ref-xml). Esencialmente significa que todos los elementos bien deben tener etiquetas de cierre bien deben ser escritos de manera especial (tal y como se describe abajo), y que todos los elementos deben estar anidados.

Aunque el solapamiento de elementos no está permitido en SGML, era tolerado en los navegadores existentes.

CORRECTO: elementos anidados

<p>he aquí un <em>párrafo</em> enfatizado.</p>

INCORRECTO: elementos solapados

<p>he aquí un <em>párrafo</p> enfatizado</em>

## 4.2 Los nombres de elementos y atributos deben escribirse en minúscula

Los documentos XHTML deben usar minúsculas para los nombres de todos los elementos y atributos HTML. Esta diferencia es necesaria porque XML es sensible a minúsculas y mayúsculas e.g. <li> and <LI> son etiquetas diferentes.

## 4.3 Los elementos no vacíos requieren etiquetas de cierre

Con HTML 4.0, basado en SGML, en algunos elementos podía omitirse la etiqueta de cierre, de tal manera que la apertura de los elementos que les sucedían implicaba dicho cierre. Esta omisión no está permitida en XHTML, basado en XML. Todos los elementos que no estén declarados en la DTD como EMPTY deben tener una etiqueta de cierre.

CORRECTO: elementos cerrados

<p>he aquí un párrafo.</p><p>aquí hay otro párrafo.</p>

INCORRECTO: elementos no cerrados

<p>he aquí un párrafo.<p>aquí hay un párrafo.

## 4.4 Los valores de los atributos deben ir entre comillas

Todos los valores de atributos deben ir entrecomillados, incluso aquellos que son numéricos.

CORRECTO: valores de atributo entrecomillados

<table rows="3">

INCORRECTO: valores de atributo no entrecomillados

<table rows=3>

## 4.5 Minimización de atributos

XML no soporta la minimización de atributos. Los pares atributo-valor deben escribirse en toda su extensión. Los nombres de atributos como compact y checked no pueden aparecen en elementos sin que sea especificado su valor.

CORRECTO: atributos no minimizados

<dl compact="compact">

INCORRECTO: atributos minimizados

<dl compact>

## 4.6 Elementos vacíos

Los elementos vacíos deben bien tener una etiqueta de cierre bien terminar su etiqueta de apertura con />. Por ejemplo, <br/> o <hr></hr>. Ver las [directrices de compatibilidad con HTML](#guidelines)para recabar información sobre cómo asegurar la compatibilidad retroactiva con aplicaciones de usuario HTML 4.0.

CORRECTO: etiquetas vacías cerradas

<br/><hr/>

INCORRECTO: etiquetas vacías no cerradas

<br><hr>

## 4.7 Manipulación de espacios en blanco dentro de los valores de atributos

En los valores de atributos, las aplicaciones de usuario eliminarán los espacios en blanco de encabezamiento y terminación y sustituirán las secuencias de uno o más espacios en blanco (incluyendo los saltos de línea) por un único espacio en blanco entre palabras (un carácter ASCII de espacio en blanco para escrituras occidentales). Ver la[sección 3.3.3](http://www.w3.org/TR/REC-xml#AVNormalize) de [[XML]](#ref-xml).

## 4.8 Elementos script y style

En XHTML, los elementos script y style se declaran como elementos con contenido #PCDATA. Como resultado, < y & serán tratados como comienzos de etiquetado, y entidades como &lt; y&amp; serán reconocidas como referencias a las entidades < y & respectivamente por el procesador XML. Englobar el contenido del elemento script o style dentro de una sección marcada como CDATA evita el procesamiento de estas entidades.

<script>

<![CDATA[

... contenido no procesado del script ...

]]>

</script>

Las secciones CDATA son reconocidas por el procesador XML y aparecen como nodos en el Modelo del Objeto Documento (DOM), ver la [sección 1.3](http://www.w3.org/TR/REC-DOM-Level-1/level-one-core.html#ID-E067D597) de la Recomendación DOM Level 1[[DOM]](#ref-dom).

Una alternativa es usar documentos externos de estilo y escritura de código.

## 4.9 Las exclusiones de SGML

SGML da al escritor de una DTD la posibilidad de impedir que elementos específicos estén anidados en otros elementos. Tales prohibiciones (denominadas "exclusiones") no son posibles de realizar en XML.

Por ejemplo, la DTD de HTML 4.0 Strict prohibe el anidamiento de un elemento 'a' dentro de otro elemento 'a' en cualquier profundidad de anidamiento. No es posible dictar tal prohibición en XML. Aunque tales prohibiciones no puedan definirse en la DTD, algunos elementos no deberían anidarse. Un resumen de dichos elementos y los elementos que no deberían anidarse en ellos se encuentra en la normativa [Apéndice B](#prohibitions).

## 4.10 Elementos con atributos id y name

HTML 4.0 definía el atributo name para los elementos a, applet, frame, iframe, img, y map. HTML 4.0 también introducía el atributo id. Ambos atributos están diseñados para ser usados como identificadores de fragmentos de información.

En XML, los identificadores de fragmentos son del tipo ID, y tan sólo puede haber un único atributo de tipo ID por elemento. Por tanto, en XHTML 1.0 el atributo id se define con tipo ID. Con objeto de asegurar que los documentos XHTML 1.0 sean documentos XML bien estructurados, los documentos XHTML 1.0 DEBEN usar el atributo id para definir un identificador de fragmento, incluso en elementos que históricamente también hayan usado el atributo name. Ver las[directrices de compatibilidad con HTML](#guidelines) para recabar información sobre cómo asegurar la compatibilidad retroactiva de los anclajes cuando se sirvan documentos XHTML cuyo tipo de soporte de internet (MIME) sea text/html.

Notar que en XHTML 1.0, el atributo name de dichos elementos está formalmente prohibido y desaparecerá en la siguiente versión de XHTML.

# 5. Compatibilidad

Aunque no hay ninguna obligación de que los documentos XHTML 1.0 sean compatibles con las aplicaciones de usuario existentes, en la práctica es algo fácil de conseguir. Las directrices para crear documentos compatibles pueden encontrarse en el [Apéndice C](#guidelines).

## 5.1 Especificación del Tipo de Soporte de Internet

En el momento de la publicación de esta recomendación, el etiquetado MIME general recomendado para aplicaciones basadas en XML aún no ha sido decidido.

Sin embargo, los documentos XHTML que sigan la directrices indicadas en el [Apéndice C](#guidelines), "Directrices de compatibilidad con HTML " pueden ser etiquetados con el tipo de soporte de internet "text/html", dado que son compatibles con la mayoría de los navegadores HTML. Este documento no hace ninguna recomendación sobre el etiquetado MIME de otros documentos XHTML.

# 6. Futuras Líneas de Actuación

XHTML 1.0 sienta la base para una familia de tipos de documentos que extenderán y acotarán XHTML con objeto de soportar un amplio rango de nuevos dispositivos y aplicaciones, definiendo módulos que especifiquen un mecanismo para combinar dichos módulos. Dicho mecanismo permitirá la extensión y el acotamiento de XHTML 1.0 de una manera uniforme a través de la definición de nuevos módulos.

## 6.1 Modularizar HTML

A la vez que el uso de XHTML vaya pasando de las aplicaciones de usuario del ordenador de sobremesa tradicional a otras plataformas, está claro que no todos los elementos de XHTML serán necesarios en todas las plataformas. Por ejemplo un dispositivo de mano o un teléfono móvil pueden soportar sólo un subconjunto elementos de XHTML.

El proceso de modularización rompe XHTML en una serie de pequeños conjuntos de elementos. Dichos elementos pueden ser recombinados para cumplir las necesidades de diferentes comunidades.

Estos módulos se definirán en un documento posterior de la W3C.

## 6.2 Subconjuntos y Extensibilidad

La modularización conlleva diversas ventajas:

* Provee un mecanismo formal para acotar XHTML.
* Provee un mecanismo formal para extender XHTML.
* Simplifica la transformación entre tipos de documento.
* Promueve la reutilización de módulos en nuevos tipos de documento.

## 6.3 Perfiles de Documento

Un perfil de documento especifica la sintáxis y la semántica de un conjunto de documentos. La conformidad con un perfil de documento provee una base para la garantía de interoperabilidad. El perfil de documento especifica los recursos necesarios para procesar los documentos de dicho tipo, e.g. qué formatos de imagen pueden usarse, niveles de escritura de codigo, soporte de hojas de estilo, etc.

Para diseñadores de productos, esto permite a distintos grupos la definición de su propio perfil estándar.

Para autores, esto permitirá obviar la necesidad de escribir diferentes versiones de documentos para diferentes clientes.

Para grupos especiales tales como químicos, médicos o matemáticos esto permitirá la construcción de un perfil especial usando elementos HTML estándar más un grupo de elementos específicamente diseñados para cubrir las necesidades de los especialistas.

# Apéndice A. DTD

**Este apéndice es normativo.**

Estas DTD y conjuntos de entidades forman una parte normativa de esta especificación. El conjunto completo de archivos DTD conjuntamente con una declaración XML y el Catálogo Abierto SGML se incluye en el [archivo zip](http://www.w3.org/TR/xhtml1/xhtml1.zip) disponible para esta especificación.

## A.1 Definiciones del Tipo de Documento

Estas DTD se aproximan a las DTD de HTML 4.0. Se trata de que cuando, en un futuro, estas DTD se modularicen, se emplee un método de construcción de DTD que se corresponda más claramente con HTML 4.0.

* [XHTML-1.0-Strict](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)
* [XHTML-1.0-Transitional](http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd)
* [XHTML-1.0-Frameset](http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd)

## A.2 Conjunto de Entidades

Los conjuntos de entidades XHTML predefinidas son los mismos que en HTML 4.0, pero han sido modificados para ser declaraciones de entidades válidas en XML 1.0. Fijémonos en que la entidad para el signo del Euro (&euro; o &#8364; o &#x20AC;) se define como una parte de los caracteres especiales.

* [Caracteres Latin-1](http://www.w3.org/TR/xhtml1/DTD/xhtml-lat1.ent)
* [Caracteres Especiales](http://www.w3.org/TR/xhtml1/DTD/xhtml-special.ent)
* [Símbolos](http://www.w3.org/TR/xhtml1/DTD/xhtml-symbol.ent)

# Apéndice B. Incompatibilidades entre elementos

**Este apéndice es normativo.**

A continuación se detallan las incompatibilidades en el anidamiento de elementos (ver la [sección 4.9](#h-4.9)). Esta prohibición se aplica a todas las profundidades de anidamiento, i.e. afecta a todos los elementos descendientes de aquel para el que se especifica la restricción.

**a**

no puede contener otros elementos a.

**pre**

no puede contener los elementos img, object, big, small, sub o sup.

**button**

no puede contener los elementos input, select, textarea, label, button, form, fieldset,iframe o isindex.

**label**

no puede contener otros elementos label.

**form**

no puede contener otros elementos form.

# Apéndice C. Directrices de Compatibilidad con HTML

**Este apéndice es informativo.**

Este apéndice resume las directrices de diseño para autores que quieren que sus documentos XHTML puedan ser presentados en aplicaciones de usuario HTML ya existentes.

## C.1 Instrucciones de Proceso

Hay que ser consciente de que las instrucciones de proceso se ejecutan en algunas aplicaciones de usuario. Sin embargo, hay que notar que cuando la declaración XML no se incluye en un documento, éste sólo puede usar las codificaciones de caracteres por defecto UTF-8 o UTF-16.

## C.2 Elementos Vacíos

Incluir un espacio en blanco antes de la barra y ángulo de cierre / y > de los elementos vacíos, e.g. <br />, <hr /> y <img src="karen.jpg" alt="Karen" />. También, usar la sintaxis minimizada de etiquetas para los elementos vacíos, e.g. <br />, dado que la sintaxis alternativa a<br></br> permitida por XML da resultados no previsibles en muchos de las aplicaciones de usuario ya existentes.

## C.3 Minimización de Elementos y Contenido de Elementos Vacíos

Dada una instancia vacía de un elemento cuyo modelo de contenido no es EMPTY (por ejemplo, un título o párrafo vacíos) no usar la forma minimizada (e.g. usar <p> </p> y no <p />).

## C.4 Hojas de Estilo y Archivos de Código Incrustados

Usar hojas de estilo externas si la hoja en cuestión utiliza los caracteres < o & o ]]> o --. Usar archivos externos de código si el código utiliza los caracteres < o & o ]]> o --. Notar que los analizadores XML tienen permitido suprimir el contenido de los comentarios. De esta manera, la práctica común hasta ahora de "esconder" los fragmentos de código (script) y hojas de estilo (style) entre comentarios, para hacerlos invisibles a antiguos navegadores, normalmente no funcionará en aplicaciones basadas en XML.

## C.5 Saltos de Línea dentro de Valores de Atributos

Evitar saltos de línea y múltiples espacios en blanco dentro de los valores de los atributos. Estos son manipulados de manera inconsistente por las aplicaciones de usuario.

## C.6 Isindex

No incluir más de un elemento isindex en el head del documento. El elemento isindex se tiende a descartar en favor del elemento input.

## C.7 Los atributos lang y xml:lang

Úsense ambos atributos, lang y xml:lang, cuando se quiera especificar el idioma de un elemento. El valor del atributo xml:lang tiene preferencia.

## C.8 Identificadores de Fragmentos

En XML, los URI [[RFC2396](#ref-rfc2396)] que terminan con identificadores de fragmentos de la forma "#foo"no se refieren a elementos con un atributo name="foo"; por el contrario se refieren a elementos con un atributo del tipo ID, e.g., el atributo id de HTML 4.0. Muchos clientes de HTML actuales no soportan este uso de atributos de tipo ID, de tal manera que se puede dar valores idénticos a ambos atributos para aseguran la máxima compatibilidad futura y retroactiva (e.g., <a id="foo" name="foo">...</a>).

Más aún, dado que el conjunto de valores permitidos para atributos del tipo ID es mucho menor que los permitidos para atributos del tipo CDATA, el tipo del atributo name ha sido cambiado aNMTOKEN. Este atributo está limitado de tal manera que sólo puede tomar los mismos valores que los de tipo ID o los de la producción Name de XML 1.0, sección 2.5, producción 5. Desafortunadamente esta limitación no puede expresarse en las DTD de XHTML 1.0. Debido a este cambio, debe tenerse cuidado cuando se conviertan documentos HTML ya existentes a XHTML 1.0. Los valores de estos atributos deben ser unicos en todo el documento, válidos, y tales que cualquier referencia a estos identificadores de fragmentos (tanto interna como externa) deben actualizarse durante la conversión.

Finalmente, notar que XHTML 1.0 tiende a desechar el atributo name de los elementos a, applet,frame, iframe, img, y map, y será eliminado en versiones posteriores de XHTML.

## C.9 Codificación de caracteres

Para especificar una codificación de caracteres en el documento, usar tanto la especificación del atributo de codificación en la declaración xml (e.g. <?xml version="1.0" encoding="EUC-JP"?>) como una sentencia meta http-equiv (e.g. <meta http-equiv="Content-type" content='text/html; charset="EUC-JP"' />). El valor del atributo de codificación de la instrucción de proceso xml tiene preferencia.

## C.10 Atributos booleanos

Algunas aplicaciones de usuario no son capaces de interpretar atributos booleanos cuando estos aparecen en su forma extendida (no minimizada), tal y como requiere XML 1.0. Notar que este problema no afecta a aplicaciones de usuario conformes a la especificación HTML 4.0. Los siguientes atributos se encuentran afectados: compact, nowrap, ismap, declare, noshade, checked,disabled, readonly, multiple, selected, noresize, defer.

## C.11 EL Modelo del Objeto Documento y XHTML

La Recomendación de nivel 1 del Modelo del Objeto Documento [[DOM](#ref-dom)] define interfaces del modelo del objeto documento para XML y HTML 4.0. El modelo del objeto documento de HTML 4.0 especifica que los nombres de los elementos y atributos HTML se devuelven en mayúsculas. El modelo del objeto documento de XML especifica que los nombres de los elementos y atributos se devuelven con el tipo en que se hayan escrito en el propio documento. En XHTML 1.0, los elementos y atributos se escriben en minúsculas. Esta diferencia aparente puede ser resuelta de dos modos:

1. Las aplicaciones que accedan a documentos XHTML servidos como tipo de soporte Internet text/html via el DOM pueden usar el DOM HTML, y asegurarse así de que los nombres de los elementos y atributos serán devueltos en mayúsculas por dichas interfaces.
2. Las aplicaciones que accedan a documentos XHTML servidos como tipo de soporte Internet text/xml o application/xml pueden usar también el DOM XML. Los elementos y atributos serán devueltos en minúsculas. Además, algunos elementos XML pueden o no aparecer en el árbol de objetos porque son opcionales en el modelo de contenidos (e.g. el elemento tbody dentro de table). Esto ocurre porque en HTML 4.0 a algunos elementos se les permitía ser minimizados de tal manera que tanto la etiqueta de apertura como la de cierre se omitían (una característica de SGML). Esto no es posible en XML. En vez de hacer obligatorios estos elementos que no solían usarse, XHTML ha optado por hacerlos opcionales. Las aplicaciones necesitan adaptarse a ello.

## C.12 Uso del carácter & en Valores de Atributos

Cuando el valor de un atributo contenga un carácter &, debe expresarse como una referencia a la entidad de tipo carácter (e.g. "&amp;"). Por ejemplo, cuando el atributo href deel elemento aapunte a un código CGI que tome parámetros , debe expresarse como http://my.site.dom/cgi-bin/myscript.pl?class=guest&amp;name=user en vez de http://my.site.dom/cgi-bin/myscript.pl?class=guest&name=user.

## C.13 Hojas de Estilo en Cascada (CSS) y XHTML

La Recomendación de nivel 2 de las Hojas de Estilo en Cascada [[CSS2](#ref-css2)] define propiedades de estilo que se aplican al árbol de análisis del documento HTML o XML. Las diferencias en el análisis producirán diversos resultados visibles o auditivos, dependiendo de los selectores usados. Las siguientes pistas reducirán este efecto en los documentos que se sirvan sin modificación como cualquiera de estos tipos de soporte:

1. Las hojas de estilo CSS para XHTML deberían usar nombres de elementos y atributos en minúsculas.
2. En las tablas, el elemento tbody será inferido por el analizador de una aplicación de usuario HTML, pero no por el analizador de una aplicación de usuario XML. Por tanto se debería añadir siempre explícitamente un elemento tbody si se hace referencia a él en un selector CSS.
3. Dentro del espacio nominal XHTML, se espera que las aplicaciones de usuario reconozcan el atributo "id" como un atributo de tipo ID. Por tanto, las hojas de estilo deberían ser capaces de continuar usando la sintaxis taquigráfica de selectores "#" incluso si la aplicación de usuario no es capaz de leer la DTD.
4. Dentro del espacio nominal XHTML, se espera que las aplicaciones de usuario reconozcan el atributo "class". Por tanto, las hojas de estilo deberían ser capaces de continuar usando la sintaxis taquigráfica de selectores ".".
5. CSS define diferentes reglas de conformidad para los documentos HTML y XML; téngase en cuenta que las reglas HTML se aplican a los documentos XHTML suministrados como HTML y las reglas XML se aplican a los documentos XHTML suministrados como XML.

# Apéndice D. Agradecimientos

**Este apéndice es informativo.**

Esta especificación ha sido escrita con la participación de los miembros del grupo de trabajo de HTML del W3C:

Steven Pemberton, CWI (HTML Working Group Chair)  
Murray Altheim, Sun Microsystems  
Daniel Austin, AskJeeves (CNET: The Computer Network through July 1999)  
Frank Boumphrey, HTML Writers Guild  
John Burger, Mitre  
Andrew W. Donoho, IBM  
Sam Dooley, IBM  
Klaus Hofrichter, GMD  
Philipp Hoschka, W3C  
Masayasu Ishikawa, W3C  
Warner ten Kate, Philips Electronics  
Peter King, Phone.com  
Paula Klante, JetForm  
Shin'ichi Matsui, Panasonic (W3C visiting engineer through September 1999)  
Shane McCarron, Applied Testing and Technology (The Open Group through August 1999)  
Ann Navarro, HTML Writers Guild  
Zach Nies, Quark  
Dave Raggett, W3C/HP (W3C lead for HTML)  
Patrick Schmitz, Microsoft  
Sebastian Schnitzenbaumer, Stack Overflow  
Peter Stark, Phone.com  
Chris Wilson, Microsoft  
Ted Wugofski, Gateway 2000  
Dan Zigmond, WebTV Networks

# Apéndice E. Referencias

**Este apéndice es informativo.**

**[CSS2]**

["Cascading Style Sheets, level 2 (CSS2) Specification"](http://www.w3.org/TR/REC-CSS2), B. Bos, H. W. Lie, C. Lilley, I. Jacobs, 12 May 1998.  
Disponible en: <http://www.w3.org/TR/REC-CSS2>

**[DOM]**

["Document Object Model (DOM) Level 1 Specification"](http://www.w3.org/TR/REC-DOM-Level-1), Lauren Wood *et al.*, 1 October 1998.  
Disponible en: <http://www.w3.org/TR/REC-DOM-Level-1>

**[HTML]**

["HTML 4.01 Specification"](http://www.w3.org/TR/1999/PR-html40-19990824), D. Raggett, A. Le Hors, I. Jacobs, 24 August 1999.  
Disponible en: <http://www.w3.org/TR/1999/PR-html40-19990824>

**[POSIX.1]**

"ISO/IEC 9945-1:1990 Information Technology - Portable Operating System Interface (POSIX) - Part 1: System Application Program Interface (API) [C Language]", Institute of Electrical and Electronics Engineers, Inc, 1990.

**[RFC2046]**

["RFC2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types"](http://www.ietf.org/rfc/rfc2046.txt), N. Freed and N. Borenstein, November 1996.  
Disponible en <http://www.ietf.org/rfc/rfc2046.txt>. Notar que este RFC deja obsoletos los RFC1521, RFC1522, y RFC1590.

**[RFC2119]**

["RFC2119: Key words for use in RFCs to Indicate Requirement Levels"](http://www.ietf.org/rfc/rfc2119.txt), S. Bradner, March 1997.  
Disponible en: <http://www.ietf.org/rfc/rfc2119.txt>

**[RFC2376]**

["RFC2376: XML Media Types"](http://www.ietf.org/rfc/rfc2376.txt), E. Whitehead, M. Murata, July 1998.  
Disponible en: <http://www.ietf.org/rfc/rfc2376.txt>

**[RFC2396]**

["RFC2396: Uniform Resource Identifiers (URI): Generic Syntax"](http://www.ietf.org/rfc/rfc2396.txt), T. Berners-Lee, R. Fielding, L. Masinter, August 1998.  
Este documento actualiza los RFC1738 y RFC1808.  
Disponible en: <http://www.ietf.org/rfc/rfc2396.txt>

**[XML]**

["Extensible Markup Language (XML) 1.0 Specification"](http://www.w3.org/TR/REC-xml), T. Bray, J. Paoli, C. M. Sperberg-McQueen, 10 February 1998.  
Disponible en: <http://www.w3.org/TR/REC-xml>

**[XMLNAMES]**

["Namespaces in XML"](http://www.w3.org/TR/REC-xml-names), T. Bray, D. Hollander, A. Layman, 14 January 1999.  
Los espacios nominales XML proveen un método simple para calificar los nombres usados en documentos XML asociándolos con espacios nominales identificados con una URI. by URI.  
Disponible en: <http://www.w3.org/TR/REC-xml-names>

## Primer documento XHTML

* [1. Introducción](#L3060)
* [2. Fichero XHTML, editor y navegador web](#L3154)
* [3. Etiquetas](#L3208)
* [4. Primer documento XHTML](#L3325)
* [5. Ejemplo](#L3626)

## 1. Introducción

XHTML significa lenguaje de marcado de hipertexto extendido.

Se utiliza para generar documentos y contenidos de [hipertexto](http://es.wikipedia.org/wiki/Hipertexto) generalmente publicados en la WEB. Es además una reformulación del lenguaje HTML que se puede jactar de ser ahora compatible con XML.

Un lenguaje de marcado nos permite dejar indicaciones (marcas) en un documento que sirven para diferenciar distintos tipos de contenidos, estructuras o secciones. Por ejemplo hay marcas (elementos, etiquetas) que pueden "marcar" qué es un título, y su lugar en una jerarquía de importancia, o advertir que una palabra debe estar enfatizada, o bien que una oración es además un enlace hacia otro recurso.

Todo esto, es posible de marcar sin ambiguedades con el correcto uso de XHTML.

Otra de sus virtudes es que está ideado para mantener una saludable separación entre el contenido y el diseño. Es decir, que uno no afecte al otro, y se puedan modificar independientemente.

A medida que avancen y conozcan la aplicación de CSS (hojas de estilo) a los documentos notarán tal separación de manera evidente.

El XHTML está diseñado para que sea funcional no sólo a los navegadores sino a varios dispositivosWEB (teléfonos móviles, portátiles, etc). A lo largo de este manual haremos uso del término agente de usuario para referirnos a estos dispositivos capaces de interpretar documentos XHTML.

Antes de seguir puedes recorrer estos enlaces en la [Wikipedia](http://es.wikipedia.org/) para tener una introducción más amplia y conocer algo sobre la historia de estos lenguajes:

* [Lenguajes de Marcado](http://es.wikipedia.org/wiki/Lenguaje_de_Marcación)
* [SGML](http://es.wikipedia.org/wiki/SGML)
* [HTML](http://es.wikipedia.org/wiki/HTML)
* [XML](http://es.wikipedia.org/wiki/XML)
* [XHTML](http://es.wikipedia.org/wiki/XHTML)
* [CSS](http://es.wikipedia.org/wiki/Hojas_de_estilo_en_cascada)
* [W3C](http://es.wikipedia.org/wiki/W3C) (World Wide Web Consortium)

NOTA: A lo largo de éste manual muchas palabras apuntarán a la enciclopedia libre [Wikipedia](http://es.wikipedia.org/) y a otros sitios, en un intento de poder ofrecer mayor información o ampliar explicaciones sobre temas y conceptos que escapan al alcance de los objetivos de este manual.

## 2. Fichero XHTML, editor y navegador web

El fichero de un documento XHTML, en sí, es un archivo de texto plano generalmente con alguna de estas extensiónes: .html, .htm, o bien .xhtml. Los agentes de usuario interpretan estos archivos y procesan el código para mostrarlos en pantalla.

Es conveniente que a medida que aparezcan ejemplos, practiquen escribiendo sus propios documentosXHTML. Pueden editar los ficheros utilizando un [editor de textos planos](http://es.wikipedia.org/wiki/Editor_de_texto) y para poder visualizar los resultados del documento un [navegador web](http://es.wikipedia.org/wiki/Navegador) (o cualquier otro agente de usuario que interpreteXHTML).

Utilicen el editor que ustedes quieran, y al guardarlo se recomienda elegir algunda de estas extensiones, para que su sistema operativo reconozca rápidamente el tipo de fichero (.html, .htm, .xhtml.)

La elección del agente de usuario es menos trivial y se presta a discusión ya que por ejemplo algunos navegadores no cumplen de manera apropiada con los estándares (recomendaciones) que la W3Cpropone y que intenta respetar este manual.

Personalmente opto por elegir [Mozilla Firefox](http://www.mozilla-europe.org/es/products/firefox/) o [Konqueror](http://www.konqueror.org/), puedes utilizar el que tú quieras pero asegurate de que cumpla minimamente con los estándares para que los ejemplos aquí presentados funcionen de manera apropiada.

## 3. Etiquetas

En la escritura de los documentos nos valdremos de etiquetas (marcas) como principal elemento de sintaxis, que para diferenciarse del contenido, están encerradas por paréntesis angulares.

Ejemplo del uso de etiquetas, con el elemento p (párrafo):

**<p>**Contenido del parrafo**</p>**

<p> marca el inicio de la sección que contendra un párrafo, </p> marca el final del párrafo. Toda etiqueta abierta debe ser cerrada escribiendose de nuevo con el agregado de una barra entre la palabra y el parentesis de angular de apertura.

Algunos elementos que no llevan contenido se nos permite escribirlos de manera tal que la apertura y el cierre se representan con una única etiqueta. El elemento br utilizado para el salto de línea, se puede escribir así <br /> que es una simplificación de la forma también válida: <br></br>.

NOTA: Es importante notar el espacio que queda entre el fin de la palabra y la barra, en la versión resumida, de esta manera se conserva compatibilidad con algunos agentes de usuario que no reconocen la etiqueta sin ese espacio.

Las etiquetas de apertura pueden contener atributos que especifican características particulares de ésta. Los atributos tienen un nombre seguido de un igual y entre comillas el valor de tal atributo.

Ejemplo del elemento a con atributos:

**<a** href="http://www.example.com"**>**Example**</a>**

La etiqueta es <a>, el atributo es href y el valor es http://www.example.com. Con esta etiqueta se construyen los vínculos. Tanto los elementos, cómo los atributos y valores, deben escribirse en letras minúsculas. Los valores de los atributos van obligatoriamente entre comillas (incluso cuando estos valores sean números).

Los elementos soportan anidamiento (con algunas condiciones que ya conoceremos).

Ejemplo de anidamiento con los elementos a e img:

**<a** href="http://www.example.com"**><img** src="ejemplo.gif"**/></a>**

El enlace que en el ejemplo anterior se lo habíamos aplicado a la palabra Example ahora se lo aplicamos a una segunda etiqueta ( img ) la cual tiene la finalidad de mostrar imagenes, por lo que que la etiqueta más externa (elemento a) convierte a la imagen en un vínculo.

## 4. Primer documento XHTML

**<?xml** version="1.0" encoding="UTF-8"**?>**  
**<!DOCTYPE**html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"**>**  
*<!--Este es un comentario y no sera tomado en cuenta por el navegador -->*  
**<html** xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es"**>**  
**<head>**  
**<title>**Titulo de la pagina**</title>**  
**</head>**  
**<body>**  
**<p>**Primer documento XHTML, es decir un, Hola mundo**</p>**  
**</body>**  
**</html>**

Analicemos el código línea a línea:

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

Es una declaración XML, dónde específicamos el atributo enconding con el valor UTF-8. Este debería ser suficiente en la mayoría de los casos. Si quieres conocer el por qué, puedes investigar más sobre: la[codificación de caracteres](http://es.wikipedia.org/wiki/Codificación_de_caracteres) y [UTF-8](http://es.wikipedia.org/wiki/UTF-8) en la Wikipedia.

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

Este es el encabezado que deberían llevar todos los documentos XHTML1.0 estríctos acordes a las[específicaciones de la W3C](http://www.w3.org/TR/xhtml1/). Es un aviso para que el agente de usuario sepa que tipo de documento va a interpretar. Para ser más específicos diremos que al ser un documento de tipo strict.dtd, los elementos utilizados serán sólo los incluídos en el lenguaje XHTML 1.0. Si utilizaramos etiquetas antiguas y declararamos el documento cómo estricto, no podríamos [validar nuestra página XHTML](http://validator.w3.org/).

NOTA: Fijarse además que hay un salto de línea a la mitad de la etiqueta, este no es interpretado y tampoco lo es en el contenido a menos que lo explicitemos con la etiqueta <br />. Simplemente se incluyen saltos de línea para hacer el código más legible.

Hay otros dos tipos de documentos que mantienen compatibilidad con HTML 4 y soporte de frames, que no tratermos en este manual, ya que prefirero explicar un uso más estrícto del lenguaje.

Declaración de documentos Transitional:

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

Declaración de documentos Frameset:

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

Como dije trataremos de concentrarnos en los documentos declarados como estríctos.

*<!--Este es un comentario y no sera tomado en cuenta por el navegador -->*

Los comentarios comienzan con <!-- y terminan con --> (los saltos de línea no los quiebran). No son tomados en cuenta por el agente de usuario, es decir no se interpretan (por lo que no se representan) y se utilizan frecuentemente para hacer aclaraciones en el código.

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

Aquí damos comienzo al documento XHTML, todo documento debe llevar esta etiqueta de apertura (<html> ) y al finalizar debemos escribir la etiqueta de cierre que en el ejemplo se encuentra al final del código ( </html> ). Además aparecen 3 atributos, el primero específica el tipo de documento, y el segundo y el tercero deben llevar como valores el código del lenguaje en el que será escrito. Para la lengua castellana este valor es es. Puedes averiguar los [códigos de otras lenguas](http://es.wikipedia.org/wiki/ISO_639-1).

**<head>**  
**<title>**Titulo de la pagina**</title>**  
**</head>**

Con la etiqueta de apertura <head>comienza lo que se llama el encabezado de las páginas web, aquí se dan ciertas específicaciones y otra META información acerca del documento que ya veremos con más detalle. Pero entre ellas está el elemento utilizado en el ejemplo, mediante las etiquetas <title></title>, que encierran el título de la página. Los navegadores suelen mostrarlo cómo título de la ventana. También es utilizada por los buscadores cuando presentan resultados de los documentos a sus usuarios.

**<body>**  
**<p>**Primer documento XHTML, es decir un, Hola mundo**</p>**  
**</body>**

El cuerpo que comienza con <body> contiene todo el contenido que será representado por el agente de usuario. En nuestro ejemplo sólo utilizamos una etiqueta más, <p> que sirve para encerrar párrafos.

NOTA: Cada elemento puede ser un elemento de bloque o de línea (block e inline respectivamente).p es un elemento de tipo block ya que rompe la línea en el lugar dónde es incluído y crea su propio bloque, agregando un salto de línea por encima y por debajo de sus etiquetas de apertura y cierre. Otros elementos, cómo el elemento strong, son inline ya que pueden insertarse sin romper la línea.

Cerrando el cuerpo (</body>) y el documento XHTML (</html>) conseguimos nuestra primer página.

## Encabezados y marcas de importancia

* [Encabezados títulos](#Encabezado)
* [Elemento strong](#Elemento)
* [Elemento em](#Elemento1)

### Encabezados títulos

Los encabezados o títulos se pueden obtener mediante 6 etiquetas diferentes: h1, h2, h3, h4, h5 y h6que tienen una jerarquía de importancia en la manera en que se representan por los navegadores. Esto significa que h1 será la etiqueta del título de mayor importancia lo que se traduce en, por ejemplo, un tamaño de letra mas grande, h2 representaría un subtítulo con letra más pequeña y así sucesivamente hasta la etiqueta h6.

Para dar un ejemplo escribiremos un documento parecido al anterior, pero mostrando los 6 títulos posibles de mayor a menor.

**<?xml** version="1.0" encoding="UTF-8"**?>**  
**<!DOCTYPE**html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"**>**  
**<html** xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es"**>**  
**<head>**  
**<title>**Encabezados y titulos**</title>**  
**</head>**  
**<body>**  
**<h1>**Encabezado con etiqueta h1**</h1>**  
**<h2>**Encabezado con etiqueta h2**</h2>**  
**<h3>**Encabezado con etiqueta h3**</h3>**  
**<h4>**Encabezado con etiqueta h4**</h4>**  
**<h5>**Encabezado con etiqueta h5**</h5>**  
**<h6>**Encabezado con etiqueta h6**</h6>**  
**</body>**  
**</html>**

Al probar el ejemplo deberíamos ver algo así:

# Encabezado con etiqueta h1

## Encabezado con etiqueta h2

### Encabezado con etiqueta h3

#### Encabezado con etiqueta h4

##### Encabezado con etiqueta h5

###### Encabezado con etiqueta h6

[VER EJEMPLO02.html](http://chulorey.googlepages.com/ejemplo02.html).

NOTA: Dependiendo del navegador los tamaños pueden variar, pero la jerarquía se mantendrá en todos.

## Elemento strong

Semánticamente strong marca contenido al que se le quiere dar más fuerza o importancia que el resto. Los navegadores suelen formatearlo en una tipografía tipo negrita, otros agentes de usuario podrían indicarlo de maneras diferentes. Es un elemento de línea.

Ejemplo:

**<p>**Superman es oriundo de **<strong>**Krypton**</strong>**, planeta actualmente desaparecido (debido a la radiacion).**</p>**

Resultado:

Superman es oriundo de Krypton, planeta actualmente desaparecido (debido a la radiacion).

## Elemento em

El elemento em agrega énfasis al contenido. Al igual que strong es un elemento de línea.

Ejemplo

**<p>**Una corriente filosofica afirma que lo que progresa es la **<em>**tecnologia**</em>**, el hombre no.**</p>**

Resultado:

Una corriente filosofica afirma que lo que progresa es la tecnologia, el hombre no.

NOTA: Usualmente los elementos con énfasis son formateados en tipografía tipo itálica.

## Enlaces, vínculos y anchor

* [1. Introducción](#L9955)
* [2. Primer link](#L9970)
* [3. Enlace a correo electrónico y otros recursos](#L10010)
* [4. Enlaces relativos y absolutos](#L10074)
* [5. Enlaces hacia anclas en la misma página](#L10104)
* [6. El atributo title](#L10185)
* [7. Atributos rel y rev](#L10224)
  + [7.1. Valor nofollow para rel](#L10284)
  + [7.2. Valor tag para rel (Technorati)](#L10326)
* [8. Código con todos los ejemplos:](#L10364)
* [9. Ejemplo](#L10768)

## 1. Introducción

Los enlaces son los que hacen que naveguemos por la Web, es decir que vayamos saltando por diferentes páginas de un mismo sitio, o de sitios externos. La etiqueta utilizada para crear un vínculo esa (anchor).

No necesariamente un enlace apuntará a una página web, puede por ejemplo apuntar a una imagen, a un servidor FTP, a cualquier otro tipo de archivo, a un correo electrónico o incluso puede apuntar a hacia otros sectores de la misma página (previamente marcando estos sectores).

## 2. Primer link

Veamos un ejemplo sencillo:

**<a** href="http://www.example.com"**>**Example**</a>**

En el navegador lo que veremos será: [Example](http://www.example.com/). La dirección http://www.example.com es el valor del atributo href, la palabra Example, encerrada entre <a> y </a>, es lo que suele llamarse el anchor text y es el contenido que será visible en el navegador.

## 3. Enlace a correo electrónico y otros recursos

Para crear un enlace hacia un correo electrónico debemos apuntar en el valor del atributo href hacia una dirección de correo electrónico, pero es importante agregar al principio de la dirección el mailto:que vendría a representar el protocolo (http://, ftp://, etc).

Ejemplo:

**<a** href="mailto:mail@example.com"**>**Correo de ejemplo**</a>**

Como en el caso anterior, veremos en nuestro navegador: [Correo de ejemplo](mailto:mail@example.com). Si prueban el enlace seguramente su navegador disparará el cliente de correo por defecto.

Si quieres saber más sobre que URLs puedes poner en el valor del atributo hrefpuedes leer el artículo de la Wikipedia, [***U****niform****R****esource****L****ocator*](http://es.wikipedia.org/wiki/URL)*.*

## 4. Enlaces relativos y absolutos

Muchas veces para enlaces entre páginas del mismo sitio y que tengan el mismo dominio, no hace falta utilizar enlaces absolutos. Por ejemplo si estamos en la pagina www.example.com/universidades.html y queremos enlazar a www.example.com/subdirectorio/colegios.html podemos crear un enlace de la siguiente manera:

**<a** href="subdirectorio/colegios.html"**>**Ver los colegios**</a>**

Hay que destacar que en este caso hacer enlaces absolutos (escribiendo la dirección completa) funcionará también.

NOTA: Algunos buscadores recomiendan utilizar enlaces absolutos siempre que sea posible, para mejorar la indexación por parte de sus motores.

## 5. Enlaces hacia anclas en la misma página

Veamos ahora cómo crear marcas (algunos las llaman anclas) y enlaces para poder navegar en la misma página, sistema muy útil para crear un índice entre documentos largos.

La marca se realiza mediante el atributo id de la etiqueta a:

**<h2><a** id="privados"/**>**Colegios Privados**</h2>**

Hemos establecido un ancla llamada privados a la altura del título Colegios Privados ahora nos hará falta colocar un enlace desde otro sector de la página que apunte hacia esa ancla. Para esto:

**<a** href="#privados"**>**Ir a Colegios Privados**</a>**

Noten el agregado del carácter numeral (#). Este se incluye ya que también podríamos construir el enlace utilizando la dirección absoluta lo que implicaría que necesitamos una separación entre las anclas y la dirección de la página, ya que la ancla tiene su propia dirección absoluta, que en este caso sería: http://www.example.com/subdirectorio/colegios.html#privados.

NOTA 1: El atributo id para marcar anclas, se puede utilizar en cualquier etiqueta, sin embargo puede que algunos navegadores no lo reconozcan como un ancla. Por eso se suele "anclar" sobre elementosa.

NOTA 2: Es posible también usar el atributo name de la misma manera que id. La cuestión se reduce a que algunos viejos navegadores, es posible que no reconozcan el atributo id.

## 6. El atributo title

La etiqueta a tiene un atributo de importante valor semántico, llamado title. El atributo title sirve para específicar una pequeña descripción del sitio que estamos enlazando.

Los navegadores suelen desplegar un cartelito con el valor de este atributo cuando el usuario señala con el mouse el enlace. Esto es de gran ayuda a los usuarios, que decidirán si seguir el enlace o no.

Se utiliza así:

**<a** href="subdirectorio/colegios.html" title="Lista de Colegios privados de la zona sur"**>**Colegios Privados**</a>**

## 7. Atributos rel y rev

Estos atributos sirven para enriquecer semánticamente a los enlaces. Sus objetivos son funcionalmente inversos.

El atributo rel, sirve para definir el tipo de relación que tiene el documento contenedor del enlace con el que éste, está apuntando. Por ejemplo podríamos utilizarlo de la siguiente manera:

**<a** href="pagina\_04.html" title="Ir a la pagina 4" rel="next"**>**pagina 4**</a>**

El enlace anterior sería muy conveniente usarlo, por ejemplo, en la página pagina\_03.html ya que estaríamos indicando semánticamente que estamos enlazando con la página que continua el contenido de la actual.

Inversamente podríamos indicar en pagina\_04.html, mediante el elemento link y su atributo rev quépagina\_03.html es el documento que precede al actual, para esto le damos el valor previus.

Vean también una [descripción detallada del uso de de rel y rev](http://www.ics.uci.edu/~ejw/authoring/draft-ietf-html-relrev-00.txt).

### 7.1. Valor nofollow para rel

Un valor del atributo rel que ha tomado mucha popularidad en este último tiempo, es el llamadonofollow. Este fue propuesto por los buscadores cómo una manera de indicarles que el enlace que lo lleva puesto no debe ser considerado en los algoritmos que calculan el posicionamiento de los resultados de búsquedas. El objetivo de poder dar tal indicación, es luchar contra [spammers](http://es.wikipedia.org/wiki/Spam) que suelen, entre otras cosas, promocionar sus webs mediante la inserción de enlaces en sitios cómo la [Wikipedia](http://es.wikipedia.org/) o en los comentarios de los [Weblogs](http://es.wikipedia.org/wiki/Blog).

Ejemplo de su uso:

**<a** href="http://www.microsoft.com" title="Sitio que vende software privativo"rel="nofollow"**>**Microsoft**</a>**

### 7.2. Valor tag para rel (Technorati)

Mencionemos ahora otro valor para rel que puede ser utilizado para agregar etiquetados (tags) a nuestras páginas, y que es mayoritariamente utilizado por los [blogs](http://es.wikipedia.org/wiki/Blog). El valor es justamente llamado: tagy su uso es alentado por sitios como [Technorati](http://www.technorati.com/) que cuentan con tecnología para analizar este tipo de enlaces y poder categorizar nuestras webs mediante la aplicación de etiquetas (tags).

Colocando la siguiente linea de código en una página web estaremos automáticamente dandole la oportunidad a Technorati de asociar tal página con la etiqueta animaciones:

**<a** href="http://www.example.com/png/animaciones" rel="tag"**>**animaciones**</a>**

## 8. Código con todos los ejemplos:

Ahora construyamos una página completa con todos estos ejemplos:

**<?xml** version="1.0" encoding="UTF-8"**?>**  
**<!DOCTYPE**html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"**>**  
**<html** xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es"**>**  
**<head>**  
**<title>**Vinculos, hipervinculos, enlaces**</title>**  
**</head>**  
**<body>**  
**<h1><a** id="arriba" **/>**Enlaces**</h1>**  
**<p><a** href="http://www.example.com" title="Apunta hacia example.com"**>**Enlace absoluto**</a></p>**  
**<p><a** href="ejemplo02.html" title="Ir al ejemplo numero 2" rel="Prev"**>**Enlace relativo**</a></p>**  
**<p><a** href="#fin" title="Ir al final de la pagina mediante el ancla"**>**Enlace hacia ancla**</a></p>**  
**<p><a** href="mailto:mail@example.com"**>**Correo de ejemplo**</a></p>**  
**<p><a** href="ftp://ftp.gnu.org" title="Enlace al ftp del sitio GNU"**>**FTP de GNU**</a></p>**  
**<p><a** href="http://www.microsoft.com" title="Sitio que vende software privativo"rel="nofollow"**>**Enlace con NoFollow**</a></p>**  
**<p><a** href="http://technorati.com/tag/xhtml" rel="tag"**>**xhtml**</a>** **</p>**  
*<!--Forzamos varios saltos de linea para notar el uso de los enlaces mediante anclas.-->*  
**<br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br/><br** **/>**  
**<br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br/><br** **/>**  
**<br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br** **/><br/><br** **/>**  
**</p>**  
**<h3><a** id="fin" **/>**Sospecho que junto a mi hay un ancla.**</h3>**  
**<p><a** href="#arriba" title="Volver a arriba"**>**Volver**</a></p>**  
**</body>**  
**</html>**

## Citas, abreviaturas y acrónimos

* [1. Citas](#Citas)
* [2. Citas con <blockquote>](#Citas1)
* [3. Citas con <cite>](#Citas2)
* [4. Abreviaturas](#Abreviatur)
* [5. Acrónimos](#nimos)
* [6. Código del ejemplo](#L246)
* [7. Ejemplo](#Ejemplo)

## 1. Citas

En muchos documentos citaremos texto desde otras fuentes. Para especificar que estamos citando algún contenido desde otra fuente disponemos de dos etiquetas que se diferencian entre sí en el modo en que se presentará el contenido citado dentro del documento.

## 2. Citas con <blockquote>

La etiqueta <blockquote>, generalmente se utiliza para contener un párrafo ( p ) con un texto citado. El contenido citado se contendrá en un bloque que usualmente el navegador lo presentará con una tabulación mayor al resto de los párrafos.

Ejemplo:

**<blockquote>**  
**<p>**  
Linus Benedict Torvalds (born December 28, 1969 in Helsinki, Finland) is a Finnish software engineer best known for  
initiating the development of Linux.  
He now acts as the project's coordinator (or Benevolent Dictator for Life).  
**</p>**  
**<p>**  
Linus was inspired by Minix (a kernel and operating system developed by Andrew Tanenbaum) to develop a capable   
UNIX-like operating system that  
could be run on a PC. Linux now also runs on many other architectures.  
**</p>**  
**</blockquote>**

Resultado:

Linus Benedict Torvalds (born December 28, 1969 in Helsinki, Finland) is a Finnish software engineer best known for initiating the development of Linux.He now acts as the project's coordinator (or Benevolent Dictator for Life).

Linus was inspired by Minix (a kernel and operating system developed by Andrew Tanenbaum) to develop a capable UNIX-like operating system thatcould be run on a PC. Linux now also runs on many other architectures.

La anterior cita, tendrá dos saltos de línea, por encima y por debajo, de la apertura y del cierre del elemento blockquote respectivamente, ya que conforma un bloque. Si queremos citar sin estos saltos de línea debemos usar el elemento cite.

## 3. Citas con <cite>

Veamos ahora como citar en la misma línea en la que estamos escribiendo, que suele utilizarce para pequeños trozos de texto.

Ejemplo:

Mi padre sobre Linus Benedict Torvalds alguna vez supo decir **<cite>**¿Quien es ese?**</cite>**

Resultado:

Mi padre sobre Linus Benedict Torvalds alguna vez supo decir ¿Quien es ese?

Al utilizar este elemento la línea no se interrumpe como sucedería con blockquote que podría resultar molesto para una cita tan corta.

## 4. Abreviaturas

Al escribir la abreviatura de una palabra podemos valernos del elemento abbr para marcarla e indicar en el atributo title qué palabra estamos abreviando, los navegadores en general se encargarán de que al apoyar el puntero del mouse sobre la abreviatura se muestre un cartel con la palabra no abreviada que indicamos.

Ejemplo:

**<abbr** title="Telefono"**>**Tel**</abbr>**

Resultado:

Tel

## 5. Acrónimos

Los acrónimos se marcan con el elemento acronym, tienen el atributo title, y generalmente, el mismo comportamiento que abbr en los navegadores. Ejemplo:

Mi facultad es parte de la **<acronym** title="Universidad Tecnologica Nacional"**>**UTN**</acronym>**Esta página sigue los estándares de la **<acronym** **xml:lang="en"** title="World Wide Web Consortium"**>**W3C**</acronym>**

Resultado:

Mi facultad es parte de la UTN. Esta página sigue los estándares de la W3C

NOTA: Fijarse que en la segunda línea introducimos el valor en para el atributo xml:lang, este le indica que el acrónimo está escrito en lengua inglesa. Generalmente todas nuestras etiquetas soportan tal atributo, sin embargo no siempre es necesario escribirlo ya que las etiquetas heredan el valor específicado del elemento <html> (u otros elementos superiores). En nuestro caso el valor puesto en el elemento html era "es" por lo que nos vemos obligados a cambiarlo en esta etiqueta específica.

## 6. Código del ejemplo

El siguiente código es el de un documento XHTML que contiene todos los ejemplos expuestos:

**<?xml** version="1.0" encoding="UTF-8"**?>**  
**<!DOCTYPE**html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"**>**  
**<html** xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es"**>**  
**<head>**  
**<title>**Vinculos, hipervinculos, enlaces**</title>**  
**</head>**  
**<body>**  
**<h1><a** href="http://manual-xhtml.blogspot.com/" title="Ir al manual"**>**Manual XHTML**</a></h1>**  
**<h2>**Citas abreviaturas y acrónimos**</h2>**  
**<h3>**Cita:**</h3>**  
**<p>**Cita con **<em>**blockquote**</em>** de la **<a** href="http://en.wikipedia.org/wiki/Linus\_Torvalds"title="ir a la fuente"**>** Wikipedia**</a>**:**</p>**  
**<blockquote>**  
**<p>**  
Linus Benedict Torvalds (born December 28, 1969 in Helsinki, Finland) is a Finnish software engineer best known for initiating the development of Linux.  
He now acts as the project's coordinator (or Benevolent Dictator for Life).  
**</p>**  
**<p>**  
Linus was inspired by Minix (a kernel and operating system developed by Andrew Tanenbaum) to develop a capable UNIX-like operating system that  
could be run on a PC. Linux now also runs on many other architectures.  
**</p>**  
**</blockquote>**  
**<h3>**Abreviatura:**</h3>**  
**<p>**Mi padre sobre Linus Benedict Torvalds alguna vez supo decir **<cite>**¿Quien es ese?**</cite></p>**  
**<p><abbr** title="Recursos Humanos"**>**RRHH**</abbr>** **<abbr** title="Telefono"**>**Tel**</abbr>** 4000-0000**</p>**  
**<h3>**Acronimo:**</h3>**  
**<p>**Mi facultad es parte de la **<acronym** title="Universidad Tecnologica Nacional"**>**UTN**</acronym></p>**  
**</body>**  
**</html>**

## Imagenes

* [1. Imagenes formatos](#formatos05)
* [2. El elemento img](#elementoimg05)
* [3. Altura y ancho](#altyancho05)
* [4. Alineación de las imagenes (con atributo align)](#alialign05)
* [5. Alineación de las imagenes (con propiedad float)](#alifloat05)
  + [5.1. float: right](#L387)
  + [5.2. float: left](#L426)

## 1. Imagenes formatos

Además de texto, podemos incorporar al contenido de nuestros documentos XHTML archivos de imagenes que pueden contener fotografías, dibujos, diagramas, etc.

Hay una seria discusión acerca de que formato de imagenes es conveniente utilizar para el empleo en la web, que incluye temas variados y complejos cómo: algoritmos de compresión de imagenes, patentes,calidad, y compatibilidad. Personalmente me inclino por el uso de las imagenes libres de patentesPNG. Perto te recomiendo que amplies el tema para que puedas decidir que tipo de imagen utilizar según el caso, en los siguientes artículos de la [Wikipedia](http://es.wikipedia.org/):

* [Formato PNG](http://es.wikipedia.org/wiki/PNG)
* [Formato JPEG](http://es.wikipedia.org/wiki/JPEG)
* [Formato GIF](http://es.wikipedia.org/wiki/GIF)

## 2. El elemento img

El elemento para insertar imagenes es img. Su principal atributo es src (de source). Este atributo lleva como valor la dirección (relativa o absoluta) de la ubicación de la imagen.

Ejemplo de uso de img:

**<img** src="tux.png" alt="Tux, el pinguino de Linux" **/>**

Resultado:

Notar la barra antes de cerrar el parentesis angular. Como ya hemos mencionado esta es necesaria ya que en XHTML todos los elementos deben cerrarse aunque no contengan contenido, por eso utilizamos esta sintaxis para abrir y cerrar el elemento en la misma etiqueta.

Además hemos puesto el valor Tux, el pinguino de Linux al atributo alt, este debe explicar en palabras el contenido de la fotografía. Los agentes de usuario basados en texto mostrarán el valor del atributo en vez de la imagen, otros agentes de usuarios para no videntes podrían leer y reproducir con un sintetizador el valor del atributo. Es muy recomendable su uso, ya que además si la imagen por cualquier motivo no puede ser mostrada, se presentará el valor de alt en lugar de la imagen.

## 3. Altura y ancho

Podemos además específicar la altura y el ancho de la imagen, mediante el uso de los atributos height ywidth respectivamente.

En el ejemplo anterior mostramos cómo incorporar una imagen a nuestro documento sin específicar estos atributos, el problema de hacerlo de ese modo, es que el agente de usuario no podrá calcular el espacio que ocupará la imagen hasta que no termine de obtenerla. Esto podría resultar en una carga más lenta del renderizado final de la página. Es muy conveniente averiguar el tamaño de nuestras imagenes en [píxeles](http://es.wikipedia.org/wiki/Píxel) y especificarlo en los atributos height y width.

Ejemplo:

**<img** src="tux.png" alt="Tux, el pinguino de Linux" height="298" width="249" **/>**

El resultado es el mismo que antes:

Otro motivo para el uso de estos atributos, podría ser que querramos un cambio en las dimensiones y no usar el tamaño de la imagen original, para tal, tengamos en cuenta las siguientes consideraciones:

Si achicamos el ancho y la altura, lo haremos sólo en la apariencia del documento. El archivo mantendrá el tamaño original, por lo que podríamos estar gastando recursos innecesariamente. Lo conveniente sería editar y reajustar el tamaño de la imagen con algún software de edición de imagenes (por cierto [The Gimp](http://www.gimp.org/) es muy bueno y es [software GNU](http://es.wikipedia.org/wiki/GNU)).

Ejemplo:

**<img** src="tux.png" alt="Tux, el pinguino de Linux" height="120" width="100" **/>**

Resultado:

El caso inverso (aumentar las dimensiones) podría lograr un efecto no deseado que es la "pixelización" (algo así como una notable perdida de calidad) de la imagen. De nuevo es recomendable conseguir imagenes apropiadas y no valerse de estos atributos para obtener imagenes más grandes.

Ejemplo:

**<img** src="tux.png" alt="Tux, el pinguino de Linux" height="479" width="400" **/>**

En el resultado noten los bordes "pixelados":

Observemos también que deberíamos mantener las proporciones al cambiar los valores originales de la imagen mediante height y width, si no deseamos deformar el aspecto original. Una buena idea podría ser sólo reducir o aumentar uno sólo de los valores, luego el agente de usuario calculará y asignará el valor correspondiente a esa proporción para el atributo no específicado (aunque para calcular el espacio debería obtener la imagen primero, con la consecuente perdida de tiempo en el renderizado final del documento).

Ejemplo:

**<img** src="tux.png" alt="Tux, el pinguino de Linux" height="100" width="300"**/>**

El resultado es la deformación de la imagen:

Con un valor ausente:

**<img** src="tux.png" alt="Tux, el pinguino de Linux" height="100" **/>**

Ahora se mantiene la proporción ya que el valor ausente de width es calculado automáticamente:

## 4. Alineación de las imagenes (con atributo align)

El elemento img también soporta el atributo align, su uso no es recomendado, y se recomienda el uso de la propiedad float de CSS. Sin embargo esta propiedad aún no es soportada por todos los navegadores, veremos ejemplos del uso del atributo align.

NOTA: Para mantener este documento acorde al estrícto uso de elementos XHTML 1.0, se mostrarán en otra página los usos del atributo align

[Uso del atributo align](http://chulorey.googlepages.com/05usodealign.html) (en otra pagina)

## 5. Alineación de las imagenes (con propiedad float)

Para acomodar las imagenes sin el uso del atributo align, podemos valernos del atributo style que permite especificar propiedades CSS de un elemento, en este caso img.

Veamos algunos ejemplos:

#### 5.1. float: right

**<p><img** src="tux.png" alt="Tux, el pinguino de Linux" height="100" style=“float:right” **/>**....**</p>**

Si en lugar de los 3 puntos ponemos algo de texto, obetendríamos este resultado:

Tux es el nombre de la mascota oficial del kernel Linux. Tux es un pequeño pingüino de aspecto risueño. La idea de que la mascota de Linux fuera un pingüino provino del mismo Linus Torvalds, creador de Linux. Según se cuenta, cuando era niño le picó un pingüino, y le resultó simpática la idea de asociar un pingüino a su proyecto. Existen dos versiones sobre el origen de su nombre. La primera sugiere que el nombre surge del hecho de que los pingüinos parecen vestir un esmoquín (en inglés tuxedo). La segunda es que las letras que componen Tux provienen de las palabras Torvalds Unix.

#### 5.2. float: left

**<p><img** src="tux.png" alt="Tux, el pinguino de Linux" height="100" style=“float:left” **/>**....**</p>**

De nuevo un resultado parecido, pero alineada la imagen en el otro extremo:

Tux es el nombre de la mascota oficial del kernel Linux. Tux es un pequeño pingüino de aspecto risueño. La idea de que la mascota de Linux fuera un pingüino provino del mismo Linus Torvalds, creador de Linux. Según se cuenta, cuando era niño le picó un pingüino, y le resultó simpática la idea de asociar un pingüino a su proyecto. Existen dos versiones sobre el origen de su nombre. La primera sugiere que el nombre surge del hecho de que los pingüinos parecen vestir un esmoquín (en inglés tuxedo). La segunda es que las letras que componen Tux provienen de las palabras Torvalds Unix.

## Listas

* [1. Introducción](#introduccion06)
* [2. Listas ordenadas y númeradas](#numeradas06)
  + [2.1. Diferentes tipos de listas ordenadas](#tiposordenadas06)
* [3. Listas desordenadas](#desordenadas06)
  + [3.1. Diferentes tipos de listas desordenadas](#tiposdesordenadas06)
* [4. Listas de definiciones](#definition06)
* [5. Listas compuestas mediante anidamiento](#anidamiento06)

## 1. Introducción

Las listas ofrecen la posibilidad de presentar cierto tipo de información de una manera útil y simple. No sólo para ordenarla, sino también entre otras cosas para jerarquizarla, o numerarla. Para estos usos,XHTML nos permite construir diferentes tipos de listas, ya sea simples, o bien compuestas utilizando anidamiento de etiquetas.

Por ofrecer un ejemplo inmediato, resaltamos que los índices mismos de este manual utilizan listas para mostrarse.

## 2. Listas ordenadas y númeradas

Al utilizar este tipo listas, los agentes de usuario se encargarán de agregar una numeración a cada item que por defecto suele ser decimal, creciente y empezando en 1.

Veamos el siguiente código:

**<ol>**  
**<li>**Correr el script llamado "borrar\_casi\_todo\_usr\_bin.sh"**</li>**  
**<li>**Correr el comando, "apt-get install emacs21"**</li>**  
**<li>**Correr el comando, "emacs".**</li>**  
**<li>**Acostumbrarse al nuevo SO, luego conquistar el mundo.**</li>**  
**</ol>**

Este código representa una lista ordenada númericamente, con instrucciones potenciales para conquistar el mundo.

Veamos cómo queda formateada:

1. Correr el script llamado "borrar\_casi\_todo\_usr\_bin.sh"
2. Correr el comando, "apt-get install emacs21"
3. Correr el comando, "emacs".
4. Acostumbrarse al nuevo SO, luego conquistar el mundo.

El elemento ol contiene los ítems de la lista que a su vez están contenidos por el elemento li. A cada ítem introducido se le asignará un número empezando por 1.

### 2.1. Diferentes tipos de listas ordenadas

Podemos utilizar el atributo type para cambiar el sistema de numeración de las listas ordenadas. Los valores soportados por type son: A, a, 1 (por defecto) y I. Sin embargo el cambio de estas propiedades son parte del diseño por lo que no sería correcto su uso en un documento estrícto. Más adelante veremos que con el uso de hojas de estilo CSS, podremos usar aún más tipos, y manteniendo la correcta separación del contenido y el diseño del documento.

Por el momento pueden ver: [ejemplos del uso del atríbuto type](http://chulorey.googlepages.com/06_listas_type.html#typeOrdenadas) en otra página.

## 3. Listas desordenadas

Si cambiamos el elemento ol por el elemento ul, obtendremos un efecto parecido, sólo que ésta vez las listas no se presentarán con ningún tipo de ordenamiento o numeración, sino con un viñetado.

Por ejemplo observemos este código:

**<ul>**  
**<li><a** href="http://www.frba.utn.edu.ar"**>**Universidad Tecnológica de Buenos Aires**</a></li>**  
**<a><a** href="http://www.uba.ar"**>**Universidad de Buenos Aires**</a></li>**  
**<li><a** href="http://www.itba.edu.ar"**>**Instituto tecnológico de buenos aires**</a></li>**  
**</ul>**

Que va a quedar representado así:

* [Universidad Tecnológica de Buenos Aires](http://www.frba.utn.edu.ar/)
* [Universidad de Buenos Aires](http://www.uba.ar/)
* [Instituto tecnológico de buenos aires](http://www.itba.edu.ar/)

Cómo ven tenemos una lista construida con ul y dónde además cada uno de sus ítems tiene anidado un enlace. No hay ningún tipo de numeración y aunque el orden en como fueron introducidos los ítems se mantiene, no tiene ninguna importancia (recordemos que no hablamos de representación sino de valor semántico, si el orden importara deberíamos usar ol).

### 3.1. Diferentes tipos de listas desordenadas

De nuevo debemos decir que específicar tipos de listas mediante el atributo type, no es recomendable. Por el momento veremos como hacerlo en otra página. Luego veremos de que manera es conveniente mediante la aplicación de hojas de estilo CSS.

Ver [tipos de listas desordenadas mediante type](http://chulorey.googlepages.com/06_listas_type.html#typeDesordenadas) en otra página.

## 4. Listas de definiciones

Con este tipo es posible construir listas de definiciones. Cada ítem de una lista de definición contiene dos secciones, la primera es el contenido (ej. una palabra, una imagen, etc) a definir, y la segunda la definición de la primera. Se construyen con el elemento dl.

Ejemplo de código:

**<dl>**  
**<dt>**p**</dt>  
<dd>**Elemento de bloque, para construir párrafos.**</dd>**  
**<dt>**strong**</dt>  
<dd>**Elemento de línea, para remarcar la importancia de un contenido.**</dd>**  
**<dt>**em**</dt>  
<dd>**Elemento de línea, para enfátizar parte del contenido.**</dd>**  
**</dl>**

La cual se representará así:

p

Elemento de bloque, para construir párrafos.

strong

Elemento de línea, para remarcar la importancia de un contenido.

em

Elemento de línea, para enfátizar parte del contenido.

Cómo sospecharán, el elemento dt marca el término a defirnir, y el elemento dd, marca la definición del anterior.

Debido a que no llevan ni viñetado ni numeración las listas de definición no tienen diferentes tipos, aunque sí podremos modificar su apariencia con la aplicación de hojas de estilo.

## 5. Listas compuestas mediante anidamiento

Podemos anidar varias listas a otras y de diferentes tipos sin restricciones. La sintaxis es sencilla. Supongamos que tenemos una lista ordenada, si quisieramos anidarle otra lista desordenada, deberíamos crear un ítem en la más externa (la ordenada) que hará de título, pero antes de cerrar este ítem con la etiqueta </li> debemos poner el código de la lista ordenada.

Ejemplo de una lista ordenada con 3 items, el primero normal, el segundo con una lista de definición anidada, y el tercero con otra lista desordenada (que a su vez tiene otro grado de anidamiento). Lo dicho:

<ol>

<li>Item normal (lista ordenada)</li>

<li>Item titulo

<dl>

<dt>Definición1</dt>

<dd>Me llaman la definición 1 :).</dd>

<dt>Definición2</dt>

<dd>Me llaman la definición 2 :).</dd>

<dt>Definición3</dt>

<dd>Me llaman la definición 3 :).</dd>

</dl>

</li>

<li>Item título con lista anidada.

<ul>

<li>Primer item (lista desordenada) segundo grado de anidamiento.</li>

<li>Segundo item (lista desordenada) segundo grado de anidamiento.</li>

<li>Item título de segundo grado.

<ul>

<li>Primer item (lista ordenada) tercer grado de anidamiento.</li>

</ul>

</li>

</ul>

</li>

</ol>

Y el resultado es:

1. Item normal (lista ordenada)
2. Item titulo

Definición1

Me llaman la definición 1 :).

Definición2

Me llaman la definición 2 :).

Definición3

Me llaman la definición 3 :).

1. Item título con lista anidada.
   * Primer item (lista desordenada) segundo grado de anidamiento.
   * Segundo item (lista desordenada) segundo grado de anidamiento.
   * Item título de segundo grado.
     + Primer item (lista ordenada) tercer grado de anidamiento.

## Texto preformateado

* [1. Elemento pre](#pre07)
* [2. Elemento code](#code07)

## 1. Elemento pre

En algunas ocasiones podríamos necesitar incluír el contenido de un archivo de texto sin perder el formato, es decir conservando los saltos de línea y el espaciado original. Para esto contamos con el elemento pre, que permite representar texto ya preformateado.

Uno de los usos más comunes de este elemento, es la inclusión de código fuente dentro de nuestros documentos. Los navegadores, suelen representar el contenido de la etiqueta pre mediante una fuente monoespaciada.

Ejemplo incluyendo código fuente:

**<pre>**

class Libro {

public:

Libro();

void establecerTitulo(char \*);

void establecerPaginas(int);

void establecerCodigo(int);

void imprime();

private:

char Titulo[40];

int Cantpaginas;

int Codigo;

};

**</pre>**

Y el resultado sería:

class Libro {

public:

Libro();

void establecerTitulo(char \*);

void establecerPaginas(int);

void establecerCodigo(int);

void imprime();

private:

char Titulo[40];

int Cantpaginas;

int Codigo;

};

NOTA: Observar que se respetan en la representación todos los espacios y tabulaciones incluídos, y que no hace falta el uso del elemento de salto de línea br.

## 2. Elemento code

Hay que tener en cuenta que pre es un elemento en bloque, por lo que no se puede incluír en una línea sin romperla. El elemento equivalente en línea es code, que obviamente no conserva el formato (ya que no puede romper la línea), pero sigue siendo una indicación semántica de la inclusión de código. Además los agentes de usuario suelen representar el contenido de code con letra monoespaciada.

Ejemplo de uso de code:

**<p>**En el ejemplo anterior incluimos el codigo fuente (en c++) de una clase llamada**<strong>**Libro**</strong>**, con   
las funciones miembro: **<code>**Libro()**</code>**, **<code>**void establecerTitulo(char \*)**</code>** y**<code>**void imprime()**</code>**   
entre otras.**</p>**

El resultado de este párrafo sería:

En el ejemplo anterior incluimos el codigo fuente (en c++) de una clase llamada Libro, con las funciones miembro: Libro(), void establecerTitulo(char \*) y void imprime() entre otras.

Cómo notaran incluímos varios etiquetados con code que no rompieron el bloque del elemento superior, párrafo (p).

## Tablas

* [1. Introducción](#intro08)
* [2. Ejemplo sencillo](#ejem08)
* [3. El atributo summary](#summary08)
* [4. El pie de la tabla (footer)](#footer08)
* [5. Elementos de una tabla gráficamente](#grafico08)

## 1. Introducción

Las tablas son una interesante herramienta para el marcado de información tabular, otros usos y abusos (como maquetación y diseño) se desaconsejan encarecidamente. De nuevo: mantener por separado, el diseño del contenido.

## 2. Ejemplo sencillo

Veamos un poco de código para empezar a hablar de tablas:

**<table>**

**<caption>**Elementos XHTML**</caption>**

**<thead>**

**<tr>**

**<th>**ELEMENTO**</th>**

**<th>**TIPO**</th>**

**<th>**SEMANTICA**</th>**

**</tr>**

**</thead>**

**<tbody>**

**<tr>**

**<td>**em**</td>**

**<td>**línea**</td>**

**<td>**mayor énfasis**</td>**

**</tr>**

**<tr>**

**<td>**strong**</td>**

**<td>**línea**</td>**

**<td>**mayor fuerza**</td>**

**</tr>**

**<tr>**

**<td>**pre**</td>**

**<td>**bloque**</td>**

**<td>**texto preformateado**</td>**

**</tr>**

**</tbody>**

**</table>**

El resultado es el siguiente:

Elementos XHTML

| ELEMENTO | TIPO | SEMANTICA |
| --- | --- | --- |
| em | línea | mayor énfasis |
| strong | línea | mayor fuerza |
| pre | bloque | texto preformateado |

NOTA: cuando pruebes este ejemplo no verás bordes, aquí son mostrados mediante una propiedadCSS que explicaré más adelante.

El elemento table da comienzo a la tabla. Dentro tenemos a caption que encierra el título que llevará la tabla.

A partir de aquí definimos dos partes principales: thead dónde se ubica el encabezado, y tbody que encierra a las celdas del cuerpo.

tr es el encargado de contener filas, el encabezado tiene una sola fila, cuyas celdas de encabezado se contienen mediante tres etiquetas del elemento th.

Luego dentro del cuerpo (tbody) tenemos 3 filas más (tr) cuyas celdas se contienen mediante el uso del elemento td.

## 3. El atributo summary

El atributo summary se puede específicar dentro del elemento table, este nos permite agregar algo de semántica, al ponerle como valor un resumen del contenido de la tabla para que pueda ser reconocido por robots (ej. buscadores) u otros agentes de usuario.

Ejemplo de su uso:

**<table** summary="Elementos del lenguaje

XHTML"**>**...

Los navegadores no suelen representar summary.

## 4. El pie de la tabla (footer)

Vimos en el ejemplo anterior como dividir la tabla en dos secciones, mediante los elementos thead ytbody.

Ahora incluyamos una tercera sección con el elemento tfoot, el cual determinará la sección del pie (al final de la tabla).

Utilizamos un código parecido al anterior, pero con la sección foot y otros datos.

**<table** summary="Comparativa de exploits entre IE 6 y Firefox"**>**

**<caption>**Comparativa: Exploits IE vs Firefox**</caption>**

**<thead>**

**<tr>**

**<th>**Año**</th>**

**<th>**Firefox**</th>**

**<th>**Internet Explorer**</th>**

**</tr>**

**</thead>**

**<tbody>**

**<tr>**

**<td>**2004**</td>**

**<td>**15**</td>**

**<td>**27**</td>**

**</tr>**

**<tr>**

**<td>**2005**</td>**

**<td>**33**</td>**

**<td>**45**</td>**

**</tr>**

**<tr>**

**<td>**2006**</td>**

**<td>**37**</td>**

**<td>**49**</td>**

**</tr>**

**</tbody>**

**<tfoot>**

**<tr>**

**<th>**Total**</th>**

**<td>**85**</td>**

**<td>**121**</td>**

**</tr>**

**</tfoot>**

**</table>**

El resultado es el siguiente:

Comparativa: Exploits IE vs Firefox

| Año | Firefox | Internet Explorer |
| --- | --- | --- |
| Total | 85 | 121 |
| 2004 | 15 | 27 |
| 2005 | 33 | 45 |
| 2006 | 37 | 49 |

NOTA: Estos datos son inventados y las sumas no fueron verificadas :P.

En estos ejemplos podríamos haber omitido marcar secciones (encabezado, cuerpo, pie) y la tabla tendría el mismo aspecto, sin embargo marcar estas secciones no sólo la dotarán de un valor semántico sino que también nos facilitará mucho la aplicación de estilos a estos grupos de celdas.

Notar también que en la sección tfoot la celda Total fue marcada con th ya que no es un dato estádistico.

## 5. Elementos de una tabla gráficamente

Quizá los maree un poco seguir la estructura de la tabla en el código, para ayudar aquí tienen un gráfico que muestra la tabla anterior visualmente y los elementos utilizados para construirla

Nota: gráfico creado con [Kivio](http://www.koffice.org/kivio/), software gratuito para el entorno [KDE](http://www.kde.org/).

## Formularios

* [1. Introducción](#intro09)
* [2. Dos métodos post y get](#postget09)
  + [2.1. Método get](#get09)
  + [2.2. Método post](#post09)
* [3. Empezando con form](#primerform09)
* [4. Elemento input](#input09)
* [5. Campos de texto](#L1552)
* [6. Botones de radio](#radio09)
* [7. Botón de comprobación checkbox](#checkbox)
* [8. Botones de envio y de reset](#botones09)
* [9. Cajas de texto](#cajas09)
* [10. Seleccionar con select](#select09)
* [11. Elemento label](#label09)
* [12. Agrupando con fieldset](#fiedlset09)

## 1. Introducción

Los formularios nos dan la posibilidad de recibir información de parte del usuario. Su utilización nos permitirá construir formularios de entrada de datos que podríamos recibir en nuestro e-mail o bien almacenar en una base de datos dentro de nuestro servidor, para luego por ejemplo, generar una página o una [cookie](http://es.wikipedia.org/wiki/Cookie) de manera dinámica.

## 2. Dos métodos post y get

Hay dos métodos de envío de información mediante formularios que trataremos aquí: post y get.

### 2.1. Método get

El método get envía información al servidor como parte de la URL. El servidor analizará la URL pedida por el agente de usuario y separará la información relevante para su posterior procesamiento.

http://www.example.com/busqueda**?**palabra=**xhtml**

El signo ? es el delimitador que nos permite separar la consulta de la URL. busqueda podría ser un programa CGI que recibirá la información posterior a ? e interpretará tal cadena para crear una variablepalabra = xhtml. En muchos casos sencillos y para enviar cádenas de pocos carácteres este método funciona bien, sin embargo para otros casos es mejor utilizar otro método de envio.

### 2.2. Método post

Supongamos que quisieramos enviar una contraseña para loguearnos dentro de un servidor. No sería conveniente que la contraseña aparezca en la URL ya que otras personas podrían ver esta información.

Generalmente en este caso correspondería utilizar el método de envio post. Este método envía la información como parte de la petición del agente de usuario al servidor web (esta petición también incluye la URL, sin embargo nuestra información no formará parte de ella).

## 3. Empezando con form

El elemento principal y obligatorio de todo formulario es form y quizá sus dos atributos más importantes son method y action. El primero específica el método de envio (get o bien post), y en el segundo pondremos la URL del recurso que contendrá la secuencia de comandos encargada de hacer algo con esa información.

Sin embargo un formulario sin otros elementos no nos sirve de mucho. Hagamos entonces un recorrido por los elementos más comunes:

## 4. Elemento input

Ahora presentaremos al elemento input, que al específicarle su atributo type nos permite construir diferentes tipos de controles.

## 5. Campos de texto

Código para construir dos campos de texto:

**<form>**

**<p>**Nombre: **<input** type="text" name="nombre"**/></p>**

**<p>**Apellido: **<input** type="text" name="apellido"**/></p>**

**</form>**

El valor text en el atributo type indica que será un campo de texto

Ahora si quisieramos enviar esta información el agente de usuario armaría cadenas de texto, con pares del tipo nombre=VALOR1 y apellido=VALOR2 dónde VALOR1 y VALOR2 es la información escrita en las casillas de texto de Nombre y Apellido respectivamente, dichos valores se asocian al valor escrito en el atributo name del input de tipo text.

El resultado es el siguiente:

Nombre:

Apellido:

Ilustremos ahora algunas características con las que podemos dotar al input

**<form>**

**<p>**

E-mail:

**<input** type="text" name="mail" readonly="readonly" value="example@example.com" **/>**

**<strong>**readonly**</strong>**=**<em>**readonly**</em>** y **<strong>**value**</strong>**=**<em>**example@example.com**</em>**.

**</p>**

**<p>**

Contraseña:

**<input** type="password" name="password" **/>**

**<strong>**type**</strong>**=**<em>**password**</em>**

**</p>**

**<p>**

Iniciales:

**<input** type="text name" name="inicial" size="5" maxlength="2" **/>**

**<strong>**size**</strong>**=**<em>**3**</em>** y **<strong>**maxlength**</strong>**=**<em>**2**</em>**

**</p>**

**<p>**

No foco:

**<input** type="text" disabled="disabled" value="jo jo" **/>**

**<strong>**disabled**</strong>**=**<em>**disabled**</em>** y **<strong>**value**</strong>**=**<em>**jo jo**</em>**

**</p>**

**</form>**

Aquí el resultado:

E-mail:  readonly=readonly y value=example@example.com.

Contraseña:  type=password

Iniciales:  size=3 y maxlength=2

No foco:  disabled=disabled y value=jo jo

Observen detenidamente como varía el comportamiento y la apariencia de los cuadros de texto según los diferentes atributos y valores específicados.

## 6. Botones de radio

Estos botones se suelen presentar en grupo y nos permite elegir una opción entre varias. Para esto se específican valores iguales para el atributo name, mientras que para el type utilizamos el valor radio.

Piquemos código:

**<form>**

**<p>**Sobre su edad:

**<input** type="radio" name="edad" value="mayor" **/>** MAYOR

**<input** type="radio" name="edad" value="menor" **/>** MENOR

**</p>**

**<p>**Su genero:

**<input** type="radio" name="genero" value="hombre" **/>** HOMBRE

**<input** type="radio" name="genero" value="mujer" **/>** MUJER

**</p>**

**</form>**

El resultado:

Sobre su edad:  MAYOR  MENOR

Su genero:  HOMBRE  MUJER

Al especificar un valor distinto para name, se crea un nuevo grupo, mientras que dentro de un mismo grupo sólo podemos elegir una sola de las opciones.

## 7. Botón de comprobación checkbox

Ahora mostremos que pasa al poner en el atributo type el valor checkbox, que convierte a input en unBotón de comprobación.

Piquemos algo de código para dos input de tipo checkbox:

**<form>**

**<p>**

**<input** name="fuma" type="checkbox" checked="checked" value="si" **/>** Fumador

**<input** name="deporte" type="checkbox" value="si" **/>** Deportista

**</p>**

**</form>**

Observen el resultado:

 Fumador  Deportista

Uno de los input lleva el atributo checked con el valor checked, de esta forma le indicamos que debe aparecer marcado por defecto ni bien se presente el formulario.

Si enviaramos esta información al servidor, se armarían pares deporte=si y fuma=si en el caso de que las dos esten marcadas.

## 8. Botones de envio y de reset

Ahora supongamos que queremos enviar esta información a algún servidor. Como dijimos existen dos metodos de envío. Según qué enviaremos y la tecnología del servidor sabremos por cual optar. Para indicarle al agente de usuario el envío de información utilizaremos un botón de envío especificando el valor submit en el atributo type.

Probemos crear un cuadro de búsqueda para [Google](http://www.google.com/):

**<form** method="get" action="http://www.google.com/search"**>**

**<p>**

**<input** name="q" type="text" **/>**

**<input** type="submit" value="Busca en Google" **/></p>**

**</form>**

La consulta a Google se formula de de esta manera http://www.google.com/search?q=cadena. Para obtener q=cadena debemos especificar el atributo name con el valor q, luego cadena será lo que se introduzca en el cuadro de búsqueda. Por último el input text enviará toda la consulta por el método get.

Vean el resultado y testeenlo:

Otro botón muy utilizado, es el de tipo reset que se encarga de deshacer todas las modificaciones que haya hecho el usuario sobre el formulario, es decir regresa todos los elementos anidados bajo el elemento form a los valores originales especificados en el documento.

Prueben cambiar el texto dentro del cuadro y luego presionen Reset

El código anterior:

**<form>**

**<input** type="text" value="Soy el original" **/>**

**<input** type="reset" **/>**

**</form>**

## 9. Cajas de texto

Las cajas de texto se suelen utilizar para incluír una cantidad de texto mayor y de varias líneas, para la cual el input de type text no sería apropiado.

A picar código:

**<form>**

**<textarea** rows="10" cols="50" **>**Quedo adentro del cuadro.**</textarea>**

**</form>**

Como resultado obtendremos:

Los atributos rows y cols, marcan la cantidad de líneas y el tamaño de ancho respectivamente, de la caja.

## 10. Seleccionar con select

Incluyendo el elemento select podremos armar una lista de selección múltiple, este elemento se combina con el elemento option para ir agregando las diferentes opciones.

Ejemplo de código:

**<form>**

**<p>**

Elija fruta:

**<select** name="fruta"**>**

**<option>**Manzana**</option>**

**<option>**Pera**</option>**

**<option>**Kiwi**</option>**

**<option>**Uva**</option>**

**</select>**

**</p>**

**</form>**

El resultado es el siguiente:

Elija fruta:

De nuevo select cuenta con el atributo name con el valor fruta.

Sí se enviara esta información a un servidor, el agente de usuario armará la cadena de carácteres con el par fruta=Pera que se enviará por get o post, suponiendo que se haya elegido la fruta Pera.

## 11. Elemento label

Este elemento nos permite asociar una cadena de texto a un control del formulario. Esto supone una cuota de semántica ya que se identifica al control con un texto y otra de accesibilidad debido a que los agentes de usuario permiten utilizar el control mediante el texto asociado.

Ejemplo de código:

**<form>**

**<p><label><input** type="checkbox" **/>** Con label**</label></p>**

**<p><input** type="checkbox" **/>** Sin label**</p>**

**</form>**

Observen el resultado:

type="checkbox" Con label

 Sin label

En principio parecen identicas, pero prueben hacer clic sobre el texto Con label, al hacerlo la casilla de la izquierda al texto se seleccionara. En cambio al hacer clic sobre Sin label no ocurre nada.

Siempre que sea posible se recomienda utilizar el elemento label

## 12. Agrupando con fieldset

El agrupamiento es un recurso intersante para estructurar mejor nuestros formularios. El elementofieldset nos da la posibilidad de agrupar diferentes secciones bajo un marco que puede llevar una descripción al anidarle otro elemento llamado legend.

Escribamos algunos controles y agrupemos bajo dos grupos:

Código:

**<form>**

**<fieldset>**

**<legend>**Información personal**</legend>**

**<label>**Nombre: **<input** type="text" **/></label>**

**<label>**Apellido: **<input** type="text" **/></label>**

**<label><input** type="checkbox" **/>** Soltero**</label>**

**</fieldset>**

**<fieldset>**

**<legend>**Información académica**</legend>**

**<label>**Universidad: **<input** type="text" **/></label>**

**<label><input** type="checkbox" **/>** Título obtenido**</label>**

**</fieldset>**

**</form>**

Resultado de lo anterior:

Información personalNombre:type="text" Apellido:class="Apple-converted-space"  Solterclass="Apple-converted-space" Información académicaUniversidad:class="Apple-converted-space"  type="checkbox" Título obtenido

## Diferencias entre HTML y XHTML

*Artículo principal:*[*Anexo:Etiquetas HTML/XHTML*](file:///C:\wiki\Anexo:Etiquetas_HTML\XHTML)

La siguiente lista muestra algunas reglas de XHTML 1.0 que lo diferencian de [HTML](file:///C:\wiki\HTML) 4.01. Muchas de estas diferencias vienen con el cambio de ser una aplicación [SGML](file:///C:\wiki\SGML) a ser una aplicación del más estricto [XML](file:///C:\wiki\XML):

* Los elementos vacíos deben cerrarse siempre:
  + Incorrecto: <br>
  + Correcto: <br></br> o <br/> o <br />  
    Nota: Cualquiera de las tres formas es válida en XHTML. Para compatibilidad según el "Apéndice C" debe usarse <br />.
* Los elementos no vacíos también deben cerrarse siempre:
  + Incorrecto: <p>Primer párrafo<p>Segundo párrafo
  + Correcto: <p>Primer párrafo</p><p>Segundo párrafo</p>
* Los elementos anidados deben tener un correcto orden de apertura/cierre (el que se abre último, debe cerrarse primero).
  + Incorrecto: <em><strong>Texto</em></strong>
  + Correcto: <em><strong>Texto</strong></em>
* Los valores de los atributos deben siempre ir encerrados entre comillas (simples o dobles).
  + Incorrecto: <td rowspan=3>
  + Correcto: <td rowspan="3">
  + Correcto: <td rowspan='3'>
* Los nombres de elementos y atributos deben ir en minúsculas.
  + Incorrecto: <A HREF="http://www.domname.com">Domname</A>
  + Correcto: <a href="http://www.domname.com">Domname</a>
* No está permitida la minimización de atributos (se usa el nombre del atributo como valor).
  + Incorrecto: <textarea readonly>Solo-lectura</textarea>
  + Correcto: <textarea readonly="readonly">Solo-lectura</textarea>
* Los atributos desaprobados en HTML 4.01 no forman parte de XHTML.
  + Incorrecto: <font color="#0000FF">Blue text</font>
  + Correcto: <span style="color: #0000FF;">Blue text</span>

### [[editar](file:///C:\w\index.php%3ftitle=XHTML&action=edit&section=4)]Reglas para DTDs estrictos

* El texto no debe ser insertado directamente en el cuerpo (dentro de la etiqueta *body*).
  + Incorrecto: <body>Texto plano</body>
  + Correcto: <body><span>Texto plano</span></body>
* No se deben insertar elementos de bloque dentro de elementos de línea.
  + Incorrecto: <em><h2>Título</h2></em>
  + Correcto: <h2><em>Título</em></h2>

# XHTML™ 1.1 - XHTML Basado en Módulos

## Recomendación del W3C del 31 de Mayo de 2001

**Esta versión:**

<http://www.w3.org/TR/2001/REC-xhtml11-20010531> ([Single file version](http://www.w3.org/TR/xhtml11/xhtml11.html), [PostScript version](http://www.w3.org/TR/xhtml11/xhtml11.ps),[PDF version](http://www.w3.org/TR/xhtml11/xhtml11.pdf), [ZIP archive](http://www.w3.org/TR/xhtml11/xhtml11.zip), or [Gzip'd TAR archive](http://www.w3.org/TR/xhtml11/xhtml11.tgz))

**Última versión:**

<http://www.w3.org/TR/xhtml11>

**Versión Previa:**

<http://www.w3.org/TR/2001/PR-xhtml11-20010406>

**Editores:**

[Murray Altheim](mailto:murray.altheim@sun.com), [Sun Microsystems](http://www.sun.com/) [Shane McCarron](mailto:s.mccarron@opengroup.org), [Applied Testing and Technology](http://www.aptest.com/)

[Copyright](http://www.w3.org/Consortium/Legal/ipr-notice-20000612#Copyright) ©2001 [W3C](http://www.w3.org/)® ([MIT](http://www.lcs.mit.edu/), [INRIA](http://www.inria.fr/), [Keio](http://www.keio.ac.jp/)), All Rights Reserved. W3C [liability](http://www.w3.org/Consortium/Legal/ipr-notice-20000612#Legal_Disclaimer), [trademark](http://www.w3.org/Consortium/Legal/ipr-notice-20000612#W3C_Trademarks), [document use](http://www.w3.org/Consortium/Legal/copyright-documents-19990405) and [software licensing](http://www.w3.org/Consortium/Legal/copyright-software-19980720) rules apply.

## Resumen

Esta recomendación define un nuevo tipo de documento XHTML que está basado en el marco de trabajo de módulos y en dichos módulos definidos en Modularización de XHTML [[XHTMLMOD](#ref_xhtml_modularization)]. El propósito de este tipo de documento es servir como base para futuros tipos de documento extendidos de la 'familia' XHTML, y proporcionar un tipo de documento consistente, con vistas al futuro, claramente separado de la funcionalidad en desuso del legado de HTML 4 [[HTML4](#ref_html4)] que fue llevada adelante en los tipos de documento XHTML 1.0 [[XHTML1](#ref_xhtml1)]. Este tipo de documento es esencialmente una reformulación de XHTML 1.0 Strict usando Módulos XHTML. Esto significa que la mayoría de las facilidades disponibles en otros tipos de documento de la Familia XHTML (e.d., XHTML Frames) no se encuentran disponibles en este tipo de documento. Estas otras facilidades están disponibles mediante módulos definidos en Modularización de XHTML, y los autores de los documentos son libres de definir tipos de documento basados en XHTML 1.1 que utilicen dichas facilidades (ver [[XHTMLMOD](#ref_xhtml_modularization)] para información sobre la creación de nuevos tipos de documento).

## Estado de este documento

Esta sección describe el estado de este documento en el momento de su publicación. Otros documentos pueden reemplazar a este documento. El último estado de las series de este documento es mantenido por el W3C.

Este documento ha sido revisado por Miembros del W3C y otras partes interesadas y ha sido avalado por el Director como una Recomendación del W3C. Es un documento estable y puede ser citado como material de referencia o referencia normativa en otro documento. El objetivo del W3C al hacer la Recomendación es llamar la atención sobre la especificación y promover su máxima difusión. Esto potencia la funcionalidad e interoperabilidad en la Web.

Este documento ha sido producido como parte de la [Actividad HTML del W3C](http://www.w3.org/MarkUp/). Las metas del [Grupo de Trabajo de HTML](http://www.w3.org/MarkUp/Group/)*(*[*de acceso exclusivo a sus miembros*](http://cgi.w3.org/MemberAccess/)*)* se discuten en la [carta del Grupo de Trabajo de HTML](http://www.w3.org/MarkUp/Group/HTMLcharter) *(*[*de acceso exclusivo a sus miembros*](http://cgi.w3.org/MemberAccess/)*)*. Para contactar por trabajos relacionados con HTML el encargado del grupo de trabajo de HTML del W3C es [Masayasu Ishikawa](mailto:mimasa@w3.org).

La discusión pública sobre HTML tiene lugar en [www-html@w3.org](mailto:www-html@w3.org) ([archivo](http://lists.w3.org/Archives/Public/www-html/)). Para subscribirse envíe un email a [www-html-request@w3.org](mailto:www-html-request@w3.org) con la palabra subscribe en la línea del asunto.

Por favor, notifique los errores en este documento a [www-html-editor@w3.org](mailto:www-html-editor@w3.org) ([archivo](http://lists.w3.org/Archives/Public/www-html-editor/)). La lista de[errores conocidos](http://www.w3.org/2001/05/REC-xhtml11-20010531-errata) en esta especificación está disponible en http://www.w3.org/2001/05/REC-xhtml11-20010531-errata.

La versión en Inglés de esta especificación es la única versión normativa. Información acerca de[traducciones de este documento](http://www.w3.org/MarkUp/translations) está disponible en http://www.w3.org/MarkUp/translations.

Una lista de las actuales Recomendaciones del W3C y otros documentos técnicos puede encontrarse en <http://www.w3.org/TR>.

# Breve Tabla de Contenidos

* 1. [Introducción](#s_intro)
* 2. [Definición de Conformidad](#s_conform)
* 3. [El Documento del Tipo XHTML 1.1](#s_doctype)
* A. [Cambios respecto a XHTML 1.0 Strict](#a_changes)
* B. [Referencias](#a_refs)
* C. [Definición del Tipo de Documento XHTML 1.1](#a_xhtml11_dtd)
* D. [Agradecimientos](#a_acks)

# Tabla Completa de Contenidos

* 1. [Introducción](#s_intro)
* 2. [Definición de Conformidad](#s_conform)
  + 2.1. [Conformidad del Documento](#doccont)
    - 2.1.1. [Documentos Estrictamente Conformes](#strict)
  + 2.2. [Conformidad de los Agentes de Usuario](#uaconf)
* 3. [El Tipo de Documento XHTML 1.1](#s_doctype)
* A. [Cambios respecto a XHTML 1.0 Strict](#a_changes)
* B. [Referencias](#a_refs)
  + B.1. [Referencias Normativas](#a_normrefs)
  + B.2. [Referencias Informativas](#a_inforefs)
* C. [Definición del Tipo de Documento XHTML 1.1](#a_xhtml11_dtd)
  + C.1. [Entrada del Catálogo Abierto SGML para XHTML 1.1](#a_xhtml11_catalog)
  + C.2. [Driver XHTML 1.1](#a_xhtml11_driver)
  + C.3. [Personalizaciones de XHTML 1.1](#a_xhtml11_customization)
* D. [Agradecimientos](#a_acks)

# 1. Introducción

Esta sección es normativa.

Con la introducción de la familia de módulos y documentos del tipo XHTML, el W3C ha contribuido a trasladar a la comunidad de desarrolladores de contenido de Internet desde los días del marcado mal formado, no-estándar hasta el mundo bien formado y válido de XML [[XML](#ref_xml)]. En XHTML 1.0, este movimiento fue moderado por la meta de proporcionar una migración fácil desde el contenido existente, basado en HTML 4 (o anterior) a XHTML y XML. Con el advenimiento de los módulos XHTML definidos en Modularización de XHTML, el W3C ha eliminado el soporte para elementos y atributos en desuso de la familia XHTML. Estos elementos y atributos tenían una funcionalidad ampliamente orientada a la presentación que se maneja mejor vía hojas de estilo o comportamientos por defecto específicos de los clientes.

En lo sucesivo, los documentos de los tipos de la familia XHTML van a estar basados en esta nueva, más estructural colección de funcionalidades. En esta especificación, el Grupo de Trabajo de HTML del W3C ha definido un tipo de documento inicial basado exclusivamente en módulos. Este tipo de documento está diseñado para ser portable a una amplia colección de dispositivos clientes, y aplicable a la mayoría de contenidos de Internet. Los desarrolladores de Contenido que basen sus contenidos sobre la funcionalidad expresada en esta especificación pueden confiar en que van a ser consistentemente portable a través de la familia de agentes de usuario de conformidad XHTML.

# 2. Definición de Conformidad

Esta sección es normativa.

Las palabras clave "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "RECOMMENDED", "MAY", y "OPTIONAL" en este documento van a ser interpretadas como en[[RFC2119]](#ref_RFC2119).

Nota del traductor:  
Dado que el enlace proporcionado por el w3c hace referencia a los términos en inglés, se ha optado por no traducirlos, con el fin de no incluir contenido adicional al de la recomendación del W3C. Para consultar el significado de dichos términos en castellano, puede dirigirse al siguiente enlace:<http://www.sidar.org/traduc/xhtml1.htm#terms>.

## 2.1. Conformidad de Documento

Esta versión de XHTML proporciona una definición de documentos XHTML estrictamente conformes, que están restringidos a elementos y atributos procedentes del espacio de nombres XHTML.

### 2.1.1. Documentos Estrictamente Conformes

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

1. El documento debe ser conforme a los apremios expresados en el [Apéndice C](#a_xhtml11_dtd).
2. El elemento raíz del documento debe ser **<html>**.
3. El elemento raíz del documento debe designar el espacio de nombres XHTML usando el atributo **xmlns** [[XMLNAMES]](#ref_xmlns). El espacio de nombres designado para XHTML es "**http://www.w3.org/1999/xhtml**".
4. 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](#a_xhtml11_dtd) 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">

Aquí hay un 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.

## 2.2. Conformidad de Agente de Usuario

Un agente de usuario conforme debe seguir todos los requerimientos de conformidad definidos en [[XHTMLMOD](#ref_xhtml_modularization)].

# 3. El Tipo de Documento XHTML 1.1

Esta sección es normativa.

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 [[XHTMLMOD](#ref_xhtml_modularization)], 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" [[XHTMLMOD](#ref_xhtml_modularization)]). Los elementos son listados aquí con propósitos informativos, pero las definiciones de "Modularización de XHTML" deberían ser consideradas definitivas. En la versión on-line de este documento, los nombres de los módulos en la lista siguiente enlazan a las definiciones de dichos módulos contenidas en la versión actual de "Modularización de XHTML".

[**Módulo de Estructura**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_structuremodule)**\***

**body, head, html, title**

[**Módulo de Texto**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_textmodule)**\***

**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**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_hypertextmodule)**\***

**a**

[**Módulo de Lista**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_listmodule)**\***

**dl, dt, dd, ol, ul, li**

[**Módulo de Objetos**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_objectmodule)

**object, param**

[**Módulo de Presentación**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_presentationmodule)

**b, big, hr, i, small, sub, sup, tt**

[**Módulo de Edición**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_editmodule)

**del, ins**

[**Módulo de Texto Bidireccional**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_bdomodule)

**bdo**

[**Módulo de Formularios**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_extformsmodule)

**button, fieldset, form, input, label, legend, select, optgroup, option, textarea**

[**Módulo de Tablas**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_tablemodule)

**caption, col, colgroup, table, tbody, td, tfoot, th, thead, tr**

[**Módulo de Imagen**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_imagemodule)

**img**

[**Módulo de Mapa de Imagen del lado Cliente**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_imapmodule)

**area, map**

[**Módulo de Mapa de Imagen del lado Servidor**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_servermapmodule)

Attribute **ismap** on **img**

[**Módulo de Eventos Intrínsecos**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_intrinsiceventsmodule)

Events attributes

[**Módulo de Metainformación**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_metamodule)

**meta**

[**Módulo de Scripting**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_scriptmodule)

**noscript, script**

[**Módulo de Hoja de Estilo**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_stylemodule)

**style** element

[**Módulo del Atributo Style**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_styleattributemodule)**En desuso**

**style** attribute

[**Módulo de Link**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_linkmodule)

**link**

[**Módulo de Base**](http://www.w3.org/TR/xhtml-modularization/abstract_modules.html#s_basemodule)

**base**

XHTML también utiliza el módulo Ruby Annotation como es definido en [[RUBY](#ref_ruby)]:

**Ruby Annotation Module**

**ruby, rbc, rtc, rb, rt, rp**

No hay definiciones adicionales requeridas por este tipo de documento. Una implementación de este tipo de documento como DTD de XML se define en el [Apéndice C](#a_xhtml11_dtd).

# A. Cambios respecto a XHTML 1.0

Este apéndice es informativo.

Este Apéndice describe las diferencias entre XHTML 1.1 y XHTML 1.0 Strict. XHTML 1.1 representa una salida tanto de HTML 4 como XHTML 1.0. Lo más significativo es la eliminación de características que estaban en desuso. En general, la estrategia es definir un lenguaje de marcado que sea rico en funcionalidad estructural, pero que delegue en las hojas de estilo para la presentación.

Las diferencias pueden resumirse como sigue:

1. En todos los elementos, el atributo lang ha sido eliminado en favor del atributo xml:lang  (como se ha definido en [[XHTMLMOD](#ref_xhtml_modularization)]).
2. En los elementos a y map, el atributo name ha sido eliminado en favor del atributo id (como se ha definido en [[XHTMLMOD](#ref_xhtml_modularization)]).
3. La colección de elementos "ruby" ha sido añadida (como se ha definido en [[RUBY](#ref_ruby)]).

# B. Referencias

Este apéndice es normativo.

## B.1. Referencias Normativas

**[HTML4]**

[HTML 4.01 Specification](http://www.w3.org/TR/1999/REC-html401-19991224), W3C Recommendation, Dave Raggett, Arnaud Le Hors, Ian Jacobs, 24 December 1999. See: http://www.w3.org/TR/1999/REC-html401-19991224

**[XHTML1]**

[XHTML 1.0: The Extensible HyperText Markup Language](http://www.w3.org/TR/2000/REC-xhtml1-20000126), W3C Recommendation, Steven Pemberton, et al., 26 January 2000. See: http://www.w3.org/TR/2000/REC-xhtml1-20000126

**[XHTMLMOD]**

[Modularization of XHTML](http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410), W3C Recommendation, Murray Altheim, et al., 10 April 2001 See: http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410

**[XML]**

[Extensible Markup Language (XML) 1.0 (Second Edition)](http://www.w3.org/TR/2000/REC-xml-20001006), W3C Recommendation, Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, 6 October 2000. See: http://www.w3.org/TR/2000/REC-xml-20001006

**[RUBY]**

[Ruby Annotation](http://www.w3.org/TR/2001/REC-ruby-20010531), W3C Recommendation, Marcin Sawicki, et al., 31 May 2001. See: http://www.w3.org/TR/2001/REC-ruby-20010531

## B.2. Referencias Informativas

**[CATALOG]**

[Entity Management: OASIS Technical Resolution 9401:1997 (Amendment 2 to TR 9401)](http://www.oasis-open.org/html/a401.htm), Paul Grosso, Chair, Entity Management Subcommittee, SGML Open, 10 September 1997. See: http://www.oasis-open.org/html/a401.htm

**[RFC2119]**

"[Key words for use in RFCs to indicate requirement levels](http://www.ietf.org/rfc/rfc2119.txt)", RFC 2119, S. Bradner, March 1997. Available at: http://www.ietf.org/rfc/rfc2119.txt

**[XMLNAMES]**

"[Namespaces in XML](http://www.w3.org/TR/1999/REC-xml-names-19990114)", W3C Recommendation, Tim Bray, Dave Hollander, Andrew Layman, 14 January 1999. Available at: http://www.w3.org/TR/1999/REC-xml-names-19990114

# C. Definición del Tipo de Documento XHTML 1.1

Este apéndice es normativo.

## C.1. Entrada del Catálogo Abierto SGML para XHTML 1.1

Esta sección contiene la definición en el formato del Catálogo Abierto SGML [[CATALOG](#ref_catalog)] del XHTML 1.1 FPI.

-- .......................................................................... ---- File catalog ............................................................ ---- XHTML 1.1 Catalog Data File Revision: @(#)xhtml11.cat 1.9 2001/04/04 SMI See "Entity Management", SGML Open Technical Resolution 9401 for detailed information on supplying and using catalog data. This document is available from OASIS at URL: <http://www.oasis-open.org/html/tr9401.html>---- .......................................................................... ---- SGML declaration associated with XHTML .................................. --OVERRIDE YESSGMLDECL "xml1.dcl"-- :::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: ---- XHTML 1.1 DTD modular driver file ....................................... ---- note that this uses the local, flattened version of the DTD. If you want your catalog to use the master version of the XHTML Modules, change the entry to reference xhtml11.dtd instead of xhtml11-flat.dtd--PUBLIC "-//W3C//DTD XHTML 1.1//EN" "xhtml11-flat.dtd"-- XHTML 1.1 framework modules .............................................. --PUBLIC "-//W3C//ENTITIES XHTML 1.1 Document Model 1.0//EN" "xhtml11-model-1.mod"-- End of catalog data ..................................................... ---- .......................................................................... --

## C.2. XHTML 1.1 Driver

Esta sección contiene el driver para la implementación del tipo de documento XHTML 1.1 como una DTD XML. Se basa en las implementaciones de módulos XHTML definidas en [[XHTMLMOD](#ref_xhtml_modularization)] y en [[RUBY](#ref_ruby)].

<!-- ....................................................................... --><!-- XHTML 1.1 DTD ........................................................ --><!-- file: xhtml11.dtd--><!-- XHTML 1.1 DTD This is XHTML, a reformulation of HTML as a modular XML application. The Extensible HyperText Markup Language (XHTML) Copyright 1998-2001 World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. Permission to use, copy, modify and distribute the XHTML DTD and its accompanying documentation for any purpose and without fee is hereby granted in perpetuity, provided that the above copyright notice and this paragraph appear in all copies. The copyright holders make no representation about the suitability of the DTD for any purpose. It is provided "as is" without expressed or implied warranty. Author: Murray M. Altheim <altheim@eng.sun.com> Revision: $Id: xhtml11.dtd,v 1.21 2001/05/29 16:37:01 ahby Exp $--><!-- This is the driver file for version 1.1 of the XHTML DTD. Please use this formal public identifier to identify it: "-//W3C//DTD XHTML 1.1//EN"--><!ENTITY % XHTML.version "-//W3C//DTD XHTML 1.1//EN" ><!-- Use this URI to identify the default namespace: "http://www.w3.org/1999/xhtml" See the Qualified Names module for information on the use of namespace prefixes in the DTD.--><!ENTITY % NS.prefixed "IGNORE" ><!ENTITY % XHTML.prefix "" ><!-- Reserved for use with the XLink namespace:--><!ENTITY % XLINK.xmlns "" ><!ENTITY % XLINK.xmlns.attrib "" ><!-- For example, if you are using XHTML 1.1 directly, use the FPI in the DOCTYPE declaration, with the xmlns attribute on the document element to identify the default namespace: <?xml version="1.0"?> <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "xhtml11.dtd"> <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"> ... </html> Revisions: (none)--><!-- reserved for future use with document profiles --><!ENTITY % XHTML.profile "" ><!-- Bidirectional Text features This feature-test entity is used to declare elements and attributes used for bidirectional text support.--><!ENTITY % XHTML.bidi "INCLUDE" ><?doc type="doctype" role="title" { XHTML 1.1 } ?><!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: --><!-- Pre-Framework Redeclaration placeholder .................... --><!-- this serves as a location to insert markup declarations into the DTD prior to the framework declarations.--><!ENTITY % xhtml-prefw-redecl.module "IGNORE" ><![%xhtml-prefw-redecl.module;[%xhtml-prefw-redecl.mod;<!-- end of xhtml-prefw-redecl.module -->]]><!ENTITY % xhtml-events.module "INCLUDE" ><!-- Inline Style Module ........................................ --><!ENTITY % xhtml-inlstyle.module "INCLUDE" ><![%xhtml-inlstyle.module;[<!ENTITY % xhtml-inlstyle.mod

PUBLIC "-//W3C//ELEMENTS XHTML Inline Style 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-inlstyle-1.mod" >

%xhtml-inlstyle.mod;]]>

<!-- declare Document Model module instantiated in framework

-->

<!ENTITY % xhtml-model.mod

PUBLIC "-//W3C//ENTITIES XHTML 1.1 Document Model 1.0//EN"

"xhtml11-model-1.mod" >

<!-- Modular Framework Module (required) ......................... -->

<!ENTITY % xhtml-framework.module "INCLUDE" ><![%xhtml-framework.module;[<!ENTITY % xhtml-framework.mod

PUBLIC "-//W3C//ENTITIES XHTML Modular Framework 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-framework-1.mod" >

%xhtml-framework.mod;]]>

<!-- Post-Framework Redeclaration placeholder ................... -->

<!-- this serves as a location to insert markup declarations

into the DTD following the framework declarations.

-->

<!ENTITY % xhtml-postfw-redecl.module "IGNORE" ><![%xhtml-postfw-redecl.module;[%xhtml-postfw-redecl.mod;<!-- end of xhtml-postfw-redecl.module -->]]><!-- Text Module (Required) ..................................... --><!ENTITY % xhtml-text.module "INCLUDE" ><![%xhtml-text.module;[<!ENTITY % xhtml-text.mod

PUBLIC "-//W3C//ELEMENTS XHTML Text 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-text-1.mod" >

%xhtml-text.mod;]]>

<!-- Hypertext Module (required) ................................. -->

<!ENTITY % xhtml-hypertext.module "INCLUDE" ><![%xhtml-hypertext.module;[<!ENTITY % xhtml-hypertext.mod

PUBLIC "-//W3C//ELEMENTS XHTML Hypertext 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-hypertext-1.mod" >

%xhtml-hypertext.mod;]]>

<!-- Lists Module (required) .................................... -->

<!ENTITY % xhtml-list.module "INCLUDE" ><![%xhtml-list.module;[<!ENTITY % xhtml-list.mod

PUBLIC "-//W3C//ELEMENTS XHTML Lists 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-list-1.mod" >

%xhtml-list.mod;]]>

<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!-- Edit Module ................................................ -->

<!ENTITY % xhtml-edit.module "INCLUDE" ><![%xhtml-edit.module;[<!ENTITY % xhtml-edit.mod

PUBLIC "-//W3C//ELEMENTS XHTML Editing Elements 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-edit-1.mod" >

%xhtml-edit.mod;]]>

<!-- BIDI Override Module ....................................... -->

<!ENTITY % xhtml-bdo.module "%XHTML.bidi;" ><![%xhtml-bdo.module;[<!ENTITY % xhtml-bdo.mod

PUBLIC "-//W3C//ELEMENTS XHTML BIDI Override Element 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-bdo-1.mod" >

%xhtml-bdo.mod;]]>

<!-- Ruby Module ................................................ -->

<!ENTITY % Ruby.common.attlists "INCLUDE" ><!ENTITY % Ruby.common.attrib "%Common.attrib;" ><!ENTITY % xhtml-ruby.module "INCLUDE" ><![%xhtml-ruby.module;[<!ENTITY % xhtml-ruby.mod

PUBLIC "-//W3C//ELEMENTS XHTML Ruby 1.0//EN"

"http://www.w3.org/TR/ruby/xhtml-ruby-1.mod" >

%xhtml-ruby.mod;]]>

<!-- Presentation Module ........................................ -->

<!ENTITY % xhtml-pres.module "INCLUDE" ><![%xhtml-pres.module;[<!ENTITY % xhtml-pres.mod

PUBLIC "-//W3C//ELEMENTS XHTML Presentation 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-pres-1.mod" >

%xhtml-pres.mod;]]>

<!-- Link Element Module ........................................ -->

<!ENTITY % xhtml-link.module "INCLUDE" ><![%xhtml-link.module;[<!ENTITY % xhtml-link.mod

PUBLIC "-//W3C//ELEMENTS XHTML Link Element 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-link-1.mod" >

%xhtml-link.mod;]]>

<!-- Document Metainformation Module ............................ -->

<!ENTITY % xhtml-meta.module "INCLUDE" ><![%xhtml-meta.module;[<!ENTITY % xhtml-meta.mod

PUBLIC "-//W3C//ELEMENTS XHTML Metainformation 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-meta-1.mod" >

%xhtml-meta.mod;]]>

<!-- Base Element Module ........................................ -->

<!ENTITY % xhtml-base.module "INCLUDE" ><![%xhtml-base.module;[<!ENTITY % xhtml-base.mod

PUBLIC "-//W3C//ELEMENTS XHTML Base Element 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-base-1.mod" >

%xhtml-base.mod;]]>

<!-- Scripting Module ........................................... -->

<!ENTITY % xhtml-script.module "INCLUDE" ><![%xhtml-script.module;[<!ENTITY % xhtml-script.mod

PUBLIC "-//W3C//ELEMENTS XHTML Scripting 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-script-1.mod" >

%xhtml-script.mod;]]>

<!-- Style Sheets Module ......................................... -->

<!ENTITY % xhtml-style.module "INCLUDE" ><![%xhtml-style.module;[<!ENTITY % xhtml-style.mod

PUBLIC "-//W3C//ELEMENTS XHTML Style Sheets 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-style-1.mod" >

%xhtml-style.mod;]]>

<!-- Image Module ............................................... -->

<!ENTITY % xhtml-image.module "INCLUDE" ><![%xhtml-image.module;[<!ENTITY % xhtml-image.mod

PUBLIC "-//W3C//ELEMENTS XHTML Images 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-image-1.mod" >

%xhtml-image.mod;]]>

<!-- Client-side Image Map Module ............................... -->

<!ENTITY % xhtml-csismap.module "INCLUDE" ><![%xhtml-csismap.module;[<!ENTITY % xhtml-csismap.mod

PUBLIC "-//W3C//ELEMENTS XHTML Client-side Image Maps 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-csismap-1.mod" >

%xhtml-csismap.mod;]]>

<!-- Server-side Image Map Module ............................... -->

<!ENTITY % xhtml-ssismap.module "INCLUDE" ><![%xhtml-ssismap.module;[<!ENTITY % xhtml-ssismap.mod

PUBLIC "-//W3C//ELEMENTS XHTML Server-side Image Maps 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-ssismap-1.mod" >

%xhtml-ssismap.mod;]]>

<!-- Param Element Module ....................................... -->

<!ENTITY % xhtml-param.module "INCLUDE" ><![%xhtml-param.module;[<!ENTITY % xhtml-param.mod

PUBLIC "-//W3C//ELEMENTS XHTML Param Element 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-param-1.mod" >

%xhtml-param.mod;]]>

<!-- Embedded Object Module ..................................... -->

<!ENTITY % xhtml-object.module "INCLUDE" ><![%xhtml-object.module;[<!ENTITY % xhtml-object.mod

PUBLIC "-//W3C//ELEMENTS XHTML Embedded Object 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-object-1.mod" >

%xhtml-object.mod;]]>

<!-- Tables Module ............................................... -->

<!ENTITY % xhtml-table.module "INCLUDE" ><![%xhtml-table.module;[<!ENTITY % xhtml-table.mod

PUBLIC "-//W3C//ELEMENTS XHTML Tables 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-table-1.mod" >

%xhtml-table.mod;]]>

<!-- Forms Module ............................................... -->

<!ENTITY % xhtml-form.module "INCLUDE" ><![%xhtml-form.module;[<!ENTITY % xhtml-form.mod

PUBLIC "-//W3C//ELEMENTS XHTML Forms 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-form-1.mod" >

%xhtml-form.mod;]]>

<!-- Legacy Markup ............................................... -->

<!ENTITY % xhtml-legacy.module "IGNORE" ><![%xhtml-legacy.module;[<!ENTITY % xhtml-legacy.mod

PUBLIC "-//W3C//ELEMENTS XHTML Legacy Markup 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-legacy-1.mod" >

%xhtml-legacy.mod;]]>

<!-- Document Structure Module (required) ....................... -->

<!ENTITY % xhtml-struct.module "INCLUDE" ><![%xhtml-struct.module;[<!ENTITY % xhtml-struct.mod

PUBLIC "-//W3C//ELEMENTS XHTML Document Structure 1.0//EN"

"http://www.w3.org/TR/xhtml-modularization/DTD/xhtml-struct-1.mod" >

%xhtml-struct.mod;]]>

<!-- end of XHTML 1.1 DTD ................................................. -->

<!-- ....................................................................... -->

## C.3. Personalizaciones de XHTML 1.1

Un Tipo de Documento de la Familia XHTML (como es XHTML 1.1) debe definir el modelo de contenido que utiliza. Esto se realiza mediante un módulo separado de modelo de contenido que es instanciado por el Marco de Trabajo Modular de XHTML. El módulo del modelo de contenido y el Driver XHTML 1.1 (arriba) trabajan juntos para personalizar las implementaciones de módulos a los requerimientos específicos de los distintos tipos de documentos. El módulo del modelo de contenido para XHTML 1.1 se define a continuación:

<!-- ....................................................................... --><!-- XHTML 1.1 Document Model Module ...................................... --><!-- file: xhtml11-model-1.mod This is XHTML 1.1, a reformulation of HTML as a modular XML application. Copyright 1998-2001 W3C (MIT, INRIA, Keio), All Rights Reserved. Revision: $Id: xhtml11-model-1.mod,v 1.13 2001/05/29 16:37:01 ahby Exp $ SMI This DTD module is identified by the PUBLIC and SYSTEM identifiers: PUBLIC "-//W3C//ENTITIES XHTML 1.1 Document Model 1.0//EN" SYSTEM "http://www.w3.org/TR/xhtml11/DTD/xhtml11-model-1.mod" Revisions: (none) ....................................................................... --><!-- XHTML 1.1 Document Model This module describes the groupings of elements that make up common content models for XHTML elements. XHTML has three basic content models: %Inline.mix; character-level elements %Block.mix; block-like elements, eg., paragraphs and lists %Flow.mix; any block or inline elements Any parameter entities declared in this module may be used to create element content models, but the above three are considered 'global' (insofar as that term applies here). The reserved word '#PCDATA' (indicating a text string) is now included explicitly with each element declaration that is declared as mixed content, as XML requires that this token occur first in a content model specification.--><!-- Extending the Model While in some cases this module may need to be rewritten to accommodate changes to the document model, minor extensions may be accomplished by redeclaring any of the three \*.extra; parameter entities to contain extension element types as follows: %Misc.extra; whose parent may be any block or inline element. %Inline.extra; whose parent may be any inline element. %Block.extra; whose parent may be any block element. If used, these parameter entities must be an OR-separated list beginning with an OR separator ("|"), eg., "| a | b | c" All block and inline \*.class parameter entities not part of the \*struct.class classes begin with "| " to allow for exclusion from mixes.--><!-- .............. Optional Elements in head .................. --><!ENTITY % HeadOpts.mix

"( %script.qname; | %style.qname; | %meta.qname;

| %link.qname; | %object.qname; )\*"

>

<!-- ................. Miscellaneous Elements .................. -->

<!-- ins and del are used to denote editing changes

-->

<!ENTITY % Edit.class "| %ins.qname; | %del.qname;" ><!-- script and noscript are used to contain scripts and alternative content--><!ENTITY % Script.class "| %script.qname; | %noscript.qname;" ><!ENTITY % Misc.extra "" ><!-- These elements are neither block nor inline, and can essentially be used anywhere in the document body.--><!ENTITY % Misc.class

"%Edit.class;

%Script.class;

%Misc.extra;"

>

<!-- .................... Inline Elements ...................... -->

<!ENTITY % InlStruct.class "%br.qname; | %span.qname;" ><!ENTITY % InlPhras.class

"| %em.qname; | %strong.qname; | %dfn.qname; | %code.qname;

| %samp.qname; | %kbd.qname; | %var.qname; | %cite.qname;

| %abbr.qname; | %acronym.qname; | %q.qname;" >

<!ENTITY % InlPres.class

"| %tt.qname; | %i.qname; | %b.qname; | %big.qname;

| %small.qname; | %sub.qname; | %sup.qname;" >

<!ENTITY % I18n.class "| %bdo.qname;" ><!ENTITY % Anchor.class "| %a.qname;" ><!ENTITY % InlSpecial.class

"| %img.qname; | %map.qname;

| %object.qname;" >

<!ENTITY % InlForm.class

"| %input.qname; | %select.qname; | %textarea.qname;

| %label.qname; | %button.qname;" >

<!ENTITY % Inline.extra "" ><!ENTITY % Ruby.class "| %ruby.qname;" ><!-- %Inline.class; includes all inline elements, used as a component in mixes--><!ENTITY % Inline.class

"%InlStruct.class;

%InlPhras.class;

%InlPres.class;

%I18n.class;

%Anchor.class;

%InlSpecial.class;

%InlForm.class;

%Ruby.class;

%Inline.extra;"

>

<!-- %InlNoRuby.class; includes all inline elements

except ruby, used as a component in mixes

-->

<!ENTITY % InlNoRuby.class

"%InlStruct.class;

%InlPhras.class;

%InlPres.class;

%I18n.class;

%Anchor.class;

%InlSpecial.class;

%InlForm.class;

%Inline.extra;"

>

<!-- %NoRuby.content; includes all inlines except ruby

-->

<!ENTITY % NoRuby.content

"( #PCDATA

| %InlNoRuby.class;

%Misc.class; )\*"

>

<!-- %InlNoAnchor.class; includes all non-anchor inlines,

used as a component in mixes

-->

<!ENTITY % InlNoAnchor.class

"%InlStruct.class;

%InlPhras.class;

%InlPres.class;

%I18n.class;

%InlSpecial.class;

%InlForm.class;

%Ruby.class;

%Inline.extra;"

>

<!-- %InlNoAnchor.mix; includes all non-anchor inlines

-->

<!ENTITY % InlNoAnchor.mix

"%InlNoAnchor.class;

%Misc.class;"

>

<!-- %Inline.mix; includes all inline elements, including %Misc.class;

-->

<!ENTITY % Inline.mix

"%Inline.class;

%Misc.class;"

>

<!-- ..................... Block Elements ...................... -->

<!-- In the HTML 4.0 DTD, heading and list elements were included

in the %block; parameter entity. The %Heading.class; and

%List.class; parameter entities must now be included explicitly

on element declarations where desired.

-->

<!ENTITY % Heading.class

"%h1.qname; | %h2.qname; | %h3.qname;

| %h4.qname; | %h5.qname; | %h6.qname;" >

<!ENTITY % List.class "%ul.qname; | %ol.qname; | %dl.qname;" ><!ENTITY % Table.class "| %table.qname;" ><!ENTITY % Form.class "| %form.qname;" ><!ENTITY % Fieldset.class "| %fieldset.qname;" ><!ENTITY % BlkStruct.class "%p.qname; | %div.qname;" ><!ENTITY % BlkPhras.class

"| %pre.qname; | %blockquote.qname; | %address.qname;" >

<!ENTITY % BlkPres.class "| %hr.qname;" ><!ENTITY % BlkSpecial.class

"%Table.class;

%Form.class;

%Fieldset.class;"

>

<!ENTITY % Block.extra "" ><!-- %Block.class; includes all block elements, used as an component in mixes--><!ENTITY % Block.class

"%BlkStruct.class;

%BlkPhras.class;

%BlkPres.class;

%BlkSpecial.class;

%Block.extra;"

>

<!-- %Block.mix; includes all block elements plus %Misc.class;

-->

<!ENTITY % Block.mix

"%Heading.class;

| %List.class;

| %Block.class;

%Misc.class;"

>

<!-- ................ All Content Elements .................. -->

<!-- %Flow.mix; includes all text content, block and inline

-->

<!ENTITY % Flow.mix

"%Heading.class;

| %List.class;

| %Block.class;

| %Inline.class;

%Misc.class;"

>

<!-- end of xhtml11-model-1.mod -->

# Guía Breve de XHTML

[[¿Qué es?](#intro)] [[¿Para qué sirve?](#uso)] [[¿Cómo funciona?](#funcionamiento)] [[Ejemplos](#ejemplo)] [[Más información](#masinfo)]

## ¿Qué es XHTML?

XHTML (Lenguaje de Marcado de Hipertexto Extensible) es una versión más estricta y limpia de [HTML](http://w3.org/MarkUp/), que nace precisamente con el objetivo de remplazar a HTML ante su limitación de uso con las cada vez más abundantes herramientas basadas en [XML](http://w3.org/XML/). XHTML extiende HTML 4.0 combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos.

## ¿Para qué sirve?

Ante la llegada al mercado de un gran número de dispositivos, XHTML surge como el lenguaje cuyo etiquetado, más estricto que HTML, va a permitir una correcta interpretación de la información independientemente del dispositivo desde el que se accede a ella. XHTML puede incluir otros lenguajes como [MathML](http://www.w3.org/TR/MathML/), [SMIL](http://www.w3.org/TR/SMIL/)o [SVG](http://w3.org/Graphics/SVG/), al contrario que HTML.

## ¿Cómo funciona?

XHTML, al estar orientado al uso de un etiquetado correcto, exige una serie de requisitos básicos a cumplir en lo que a código se refiere. Entre estos requisitos básicos se puede mencionar una estructuración coherente dentro del documento donde se incluirían elementos correctamente anidados, etiquetas en minúsculas, elementos cerrados correctamente, atributos de valores entrecomillados, etc.

## Ejemplos

A continuación se pueden ver algunos ejemplos de los aspectos más importantes a tener en cuenta a la hora de utilizar XHTML.

* Los documentos deben estar bien formados:

Un formato correcto en un documento XHTML es muy importante. Esto quiere decir que todos los elementos deben tener etiquetas de cierre, deben estar escritos de una forma determinada y además todos los elementos deben estar anidados correctamente.

Código de elementos anidados:

**<p>**Ejemplo de elementos bien **<em>**anidados**</em>**.**</p>**

**<p>**Ejemplo de elementos mal **<em>**anidados**</p>**.**</em>**

* Los nombres de atributos y elementos deben ir en minúsculas:

Tanto los elementos como los atributos deben ir en minúsculas para todos los elementos HTML y los nombres de atributos. Esto es importante ya que XML interpreta las mayúsculas y las minúsculas de forma diferente.

**<body>**Ejemplo correcto**</body>**

**<BODY>**Ejemplo incorrecto**</BODY>**

* Los elementos que no estén vacios necesitan etiquetas de cierre:

**<p>**Ejemplo correcto.**</p>**

**<p>**Ejemplo correcto.**</p>**

**<p>**Ejemplo incorrecto.**<p>**Ejemplo incorrecto.**</p>**

* Los valores de las etiquetas deben ir siempre entre comillas:

Todos los valores de los atributos deben ir entre comillas, incluso aquellos que sean numéricos.

**<table rows="3">**

**<table rows=3>** ejemplo incorrecto

* Existen varias versiones de XHTML (1.0, 1.1, Básico, etc.). Para utilizar una versión concreta, se debe incluir antes del elemento **html** del código de la página Web, la Definición del Tipo de Documento (DTD) que se pretende utilizar. Por ejemplo, un documento en XHTML 1.1 podría especificarse de la siguiente forma:

**<?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="es">**

**<head>**

**<title>Título</title>**

**</head>**

**<body>**

**.**

**.**

**.**

**</body>**

**</html>**

## Más información sobre XHTML

[Página Principal de XHTML](http://w3.org/MarkUp/)

Toda la información sobre el Lenguaje Extensible de Etiquetado de HiperTexto.

[Preguntas frecuentes sobre HTML y XHTML](http://www.w3.org/MarkUp/2004/xhtml-faq). Disponibles versiones [en español](http://www.w3c.es/Traducciones/es/Markup/2004/xhtml-faq.htm) y [català](http://css.artnau.com/traduccions/w3c/xhtml-faq-ca.html).

Respuestas a dudas frecuentes sobre HTML y XHTML.

[Recomendaciones (X)HTML](http://www.w3.org/MarkUp/#recommendations)

Especificaciones de HTML y XHTML del W3C.

[Validador de XHTML](http://validator.w3.org/)

Servicio del W3C online y gratuito que nos permite comprobar la validez de los documentos (X)HTML.