

Estándares y accesibilidad en el diseño de interfaces Web



MANUAL DE XHTML Y CSS

Maira Vykus

Estándares y accesibilidad en el diseño de interfaces Web

MANUAL DE XHTML Y CSS



MANUAL DE XHTML Y CSS

Estándares y accesibilidad en el diseño de interfaces Web

TRABAJO FINAL DE GRADUACIÓN

Maira Vykus | DGR403
Córdoba 2007



Tabla de Contenidos



TABLA DE CONTENIDOS	4
¿CÓMO INTERPRETAR ESTE MANUAL?	10
MANUAL DE XHTML	
1. REPRESENTACIÓN DE DOCUMENTOS XHTML	13
El Conjunto de Caracteres del Documento	13
Codificaciones de caracteres	14
Referencias de caracteres	15
Caracteres no representables	16
2. INTERPRETAR SGML Y EL DTD	17
Introducción al SGML	17
Estructuras SGML usadas en XHTML	17
Cómo leer el DTD de HTML	19
3. DATOS BÁSICOS	22
Tipos de datos para elementos o atributos	22
Tabla de elementos	27
Atributos genéricos por categorías	30
4. ESTRUCTURA DE UN DOCUMENTO	40
Introducción a la estructura de un documento HTML	40
Elemento DOCTYPE (versión de XHTML)	41

TABLA DE CONTENIDOS

Elemento HTML	42
Elemento HEAD	42
Elemento TITLE	44
Elemento META	44
Elemento BODY	46
Elemento DIV	47
Elemento SPAN	48
Elementos H1, H2, H3, H4, H5, H6 (encabezados)	49
Elemento ADDRESS	50
5. TEXTO	51
Elemento P (párrafo)	51
Elemento BR (salto de línea)	52
Elemento BLOCKQUOTE (cita larga)	53
Elemento Q (cita corta)	55
Elemento EM	56
Elemento STRONG	56
Elemento CITE	57
Elemento DFN	58
Elemento CODE	58
Elemento SAMP	59
Elemento KBD	60
Elemento VAR	60
Elemento ABBR	61
Elemento ACRONYM	62
Elemento PRE	62
Elementos INS y DEL	63
Elementos SUB y SUP	65
Elemento BDO (anulación del algoritmo bidireccional)	65
Separación de palabras	66
Espacio en blanco	67
6. LISTAS	68
Lista no ordenada (ul), lista ordenada (ol) y elementos de lista (li)	68
Lista de definición (dl, dt, dd)	71
Ejemplo de lista ordenadas, no ordenadas y de definición anidadas.	72
7. TABLAS	74
Elemento TABLE	75
Elemento CAPTION	76
Elementos THEAD, TFOOT y TBODY	77
Elemento COLGROUP	79
Elemento COL	81
Elemento TR	82
Elementos TH y TD	83
Atributos de tablas	89
Ejemplo de tabla completa	95
8. VÍNCULOS	98

TABLA DE CONTENIDOS

Identificador Universal de Recursos (uri)	98
Elemento A (vínculos en body)	100
Elemento LINK (vínculos en head)	104
Elemento BASE (ruta de acceso)	105
9. OBJETOS, IMÁGENES Y APLICACIONES	107
Introducción a los objetos, imágenes y aplicaciones	107
Elemento IMG	108
Elemento OBJECT	109
Elemento PARAM	113
Mapas de imágenes	115
10. HOJAS DE ESTILO	120
Introducción a las hojas de estilo	120
Hojas de estilo incrustadas	122
Hojas de estilo externas	125
Hojas de estilo en Cascada	127
11. FUENTES Y SEPARADORES HORIZONTALES	128
Elementos TT, I, B, BIG, SMALL	128
Elemento HR	129
12. MARCOS	130
Introducción a los marcos	130
Disposición de los marcos	131
Información sobre el marco destino	138
Contenido alternativo	140
Marcos en línea	141
13. FORMULARIOS	143
Introducción a los formularios	143
Controles	143
Elemento FORM	145
Elemento INPUT	147
Elemento BUTTON	152
Elemento SELECT	153
Elemento OPTGROUP	154
Elemento OPTION	154
Elemento TEXTAREA	156
Elemento LABEL	157
Elemento FIELDSET	159
Elemento LEGEND	159
Controles deshabilitados y de sólo lectura	161
Envío de formularios	161
14. SCRIPTS	164
Introducción a los scripts	164
Diseño de documentos para agentes de usuario que soporten scripts	165
Diseño de documentos para agentes de usuario que no soporten scripts	168

MANUAL DE CSS

1. INTERPRETACIÓN DEL MANUAL DE CSS	171
Definición de una propiedad CSS	171
Propiedades resumidas	173
2. SINTAXIS Y TIPOS DE DATOS BÁSICOS	174
Sintaxis	174
Llamado a un estilo	179
Aplicación	181
Reglas frente a errores	181
Valores	182
Codificación y caracteres	187
3. SELECTORES	188
Equivalencia de patrones	188
Sintaxis de los selectores	189
4. ASIGNACIÓN DE VALORES, CASCADA, Y HERENCIA	204
Valores especificados, computados, usados y reales	204
Herencia	205
La regla @import	206
Cascada	207
5. TIPOS DE MEDIOS	210
Introducción a los tipos de medios	210
Especificación de hojas de estilo dependientes de los medios	210
Tipos de medios reconocidos	211
6. MODELO DE CAJAS	213
Dimensiones de la caja	213
Ejemplo de márgenes, rellenos y bordes	214
Propiedades del margen (margin)	216
Propiedades del relleno (padding)	217
Propiedades del Borde (border)	218
7. MODELO DE FORMATO VISUAL	223
Introducción al modelo de formato visual	223
Control de la generación de cajas	224
Posicionamiento	227
Relaciones entre 'display', 'position', y 'float'	238
Capas ('z-index')	239
Dirección del texto ('direction' y 'unicodebidi')	240
8. DETALLES DEL MODELO DE FORMATO VISUAL	242
Definición de "bloque de contención"	242
Ancho del contenido ('width')	243
Ancho mínimo y máximo ('min-width' y 'max-width')	243

TABLA DE CONTENIDOS

Altura del contenido ('height')	244
Alto mínimo y máximo ('min-height' y 'max-height')	245
Altura de la línea ('line-height' y 'vertical-align')	246
9. EFECTOS VISUALES	249
Desbordamiento ('overflow')	249
Recorte ('clip')	251
Visibilidad ('visibility')	252
10. CONTENIDO GENERADO, NUMERACIÓN AUTOMÁTICA Y LISTAS	254
Pseudo-elementos :before y :after	254
La propiedad 'content'	255
Comillas	256
Contadores y numeración automática	257
Listas	259
11. MEDIOS PAGINADOS	263
Cajas de página (regla @page)	263
Salto de página	265
Cascada en el contexto de página	266
12. COLORES Y FONDOS	267
Color del primer plano ('color')	267
Fondo	267
13. FUENTES	272
Familia de fuentes ('font-family')	272
Estilos de fuente ('font-style')	273
Small-caps ('font-variant')	274
Grosor de la fuente ('font-weight')	274
Tamaño de la fuente ('font-size')	275
Propiedad abreviada de fuente ('font')	276
14. TEXTO	278
Sangría ('text-indent')	278
Alineación ('text-align')	279
Decoración ('text-decoration')	279
Espaciado de letras ('letter-spacing')	280
Espaciado de palabras ('word-spacing')	280
Capitalización ('text-transform')	281
Espacios en blanco ('white-space')	281
15. TABLAS	283
Introducción a las tablas	283
El modelo de tabla de CSS	283
Columnas	284
Las tablas en el modelo de formato visual	285
Composición visual del contenido de las tablas	286
Bordes	289

TABLA DE CONTENIDOS

16. INTERFAZ DE USUARIO	293
Cursores ('cursor')	293
Colores Del Sistema CSS2	294
Preferencias de fuentes del usuario	295
Contornos dinámicos ('outline')	295

ANEXOS

1. INFORMACIÓN DE REFERENCIA DE SGML PARA XHTML 1.0	298
Definiciones del Tipo de Documento	298
Validación de documentos	298
Entidades de caracteres	299
2. DIRECCIONALIDAD	300
Introducción al algoritmo bidireccional	300
Herencia de la información sobre la dirección del texto	301
Especificación de la dirección del texto incluido	301
Efecto de las hojas de estilo en la bidireccionalidad	301
3. TABLAS Y AGENTES DE USUARIO NO VISUALES	302
Asociación de información de encabezado con celdas de datos	302
Categorización de celdas	304
4. HOJA DE ESTILO POR DEFECTO PARA HTML 4.0	308
5. TABLA DE ELEMENTOS	310
6. TABLA DE ATRIBUTOS	317
7. TABLA DE PROPIEDADES CSS	326

GLOSARIO

BIBLIOGRAFÍA

Referencias bibliográficas	342
Referencias Web	342

¿Cómo interpretar este manual?



El presente *Manual de XHTML y CSS* es un complemento del *Trabajo Final de Graduación* titulado “*Estándares y accesibilidad en el diseño de interfaces Web*”, cuyo objetivo es exponer las ventajas de la aplicación de ambos conceptos, es decir, la creación de interfaces Web que serán accesibles para todos los individuos, en cualquier dispositivo y sin importar el contexto, y que permitirán a los usuarios, clientes y diseñadores, obtener beneficios técnicos, económicos y funcionales.

Para este fin se desarrolla una introducción a Internet, la World Wide Web y el hipertexto, que son básicamente los responsables del surgimiento del diseño Web. Asimismo, se explican los términos *accesibilidad*, *estándares Web*, *XHTML* y *CSS*, se brindan *recomendaciones para el diseño Web* y se definen las nociones del *diseño gráfico* que se extrapolan a este nuevo medio.

Es recomendable y necesaria una lectura previa de dicho material, especialmente de los apartados dedicados a XHTML y CSS, ya que allí se introducen estos lenguajes, sus beneficios e inconvenientes y las formas de aplicarlos.

En relación al presente manual, hay ciertos puntos a tener en cuenta para interpretarlo correctamente:

Palabras claves

A lo largo de este manual se encontrarán ciertas frases o palabras clave que deben ser comprendidas en cuanto a su significado y contexto de aplicación, para lo cual se recomienda una lectura previa de los términos definidos en el Glosario. En el texto dichas términos se encuentra seguidos del signo asterisco (*).

Ejemplos y DTD

La mayoría de los ejemplos no comienzan con la declaración del tipo de documento que es obligatoria al principio de todo documento HTML/XHTML. Sin embargo, estos están basados en la definición del tipo de documento estricta, a menos que el ejemplo en cuestión se refiera a elementos o atributos definidos sólo en DTD's transicional o de marcos.

[¿CÓMO INTERPRETAR ESTE MANUAL?](#)

Siempre que se brinde el vínculo para la visualización de un ejemplo, recomendamos abrir el mismo en diferentes navegadores (Explorer, Mozilla/Firefox, Netscape, Opera, etc.) para notar las diferencias de presentación de cada uno de ellos frente a diferentes códigos XHTML o CSS.

Elementos, atributos y propiedades

Los nombres de los **elementos** del lenguaje del documento (XHTML) están en letras **mayúsculas** en los textos explicativos, y no así en los códigos de ejemplo. Si bien los elementos no pueden ser escritos en mayúsculas en XHTML, se ha adoptado esta convención para facilitar su identificación. Por ejemplo: **TABLE, BODY, TD, P**.

Los nombres de los **atributos** del lenguaje del documento están en letras **minúsculas**. Por ejemplo: **style, id, class**.

Las **propiedades y valores** CSS, los descriptores y los nombres de pseudo-clase están en **minúsculas** y son delimitados por **comillas simples** (apóstrofes). Por ejemplo: `'color', 'background', 'font'`.



Manual de XHTML

EXTENSIBLE HYPERTEXT MARKUP LANGUAGE

XHTML es el acrónimo inglés de **eXtensible HyperText Markup Language** (lenguaje extensible de marcado de hipertexto), es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas Web. Tiene las mismas funcionalidades que éste último, pero cumple especificaciones más estrictas. Su objetivo es lograr una Web semántica, donde la información, y la forma de presentarla estén claramente separadas.

XHTML es la reformulación de **HTML 4** como una aplicación **XML**.

1. Representación de documentos XHTML



EL CONJUNTO DE CARACTERES DEL DOCUMENTO

Hay ciertos caracteres abstractos que pueden formar parte de un documento HTML, entre los que se incluye la letra latina "A", la letra cirílica "I", el carácter chino que significa "agua", etc.

Para promover la interoperabilidad, XHTML debe especificar su conjunto de caracteres del documento, y estos conjuntos consisten en:

- **Un Repertorio:** Un conjunto de caracteres abstractos, tales como la letra latina "A", la letra cirílica "I", el carácter chino que significa "agua", etc.
- **Posiciones de códigos:** Un conjunto de referencias enteras a los caracteres del repertorio.

Cada documento XHTML es una secuencia de caracteres del repertorio. Los sistemas informáticos identifican cada carácter según la posición de su código; por ejemplo, en el conjunto de caracteres ASCII, las posiciones de los códigos 65, 66 y 67 se refieren a los caracteres 'A', 'B' y 'C' respectivamente.

El conjunto de caracteres **ASCII** es limitado para Web, por ello HTML usa un conjunto de caracteres más completo llamado **Conjunto Universal de Caracteres** (Universal Character Set, UCS o ISO10646), que es equivalente carácter por carácter a **Unicode**⁽¹⁾.

Ahora sólo se utilizará **[ISO10646]** para las referencias al conjunto de caracteres del documento, mientras que **Unicode** se reservará para las referencias relacionadas con el algoritmo bidireccional.

A pesar de todo, el conjunto de caracteres del documento no es suficiente para permitir a los agentes de usuario interpretar correctamente los documentos HTML codificados como una secuencia de bytes en un fichero o durante una transmisión en red.

1 Unicode (<http://www.unicode.org/unicode/standard/versions/>)

CODIFICACIONES DE CARACTERES

"**Charset**" identifica una codificación de caracteres, que **es un método para convertir una secuencia de bytes en una secuencia de caracteres a ser mostrados**. En esta conversión los servidores envían los documentos HTML a los agentes de usuario como un flujo de bytes y estos los interpretan como un flujo de caracteres.

Elección de una codificación

La codificación de documentos puede ser establecida por las herramientas de creación (Ej. editores de texto) según las convenciones usadas por el software del sistema. Estas herramientas pueden emplear cualquier codificación que cubra convenientemente la mayor parte de los caracteres contenidos en el documento. Si bien algunos caracteres pueden no abarcarse por la codificación, serán representados mediante referencias de caracteres, las cuáles hacen referencia al conjunto de caracteres y no a la codificación de caracteres.

Los servidores y los proxies pueden cambiar la codificación de caracteres (hacer una transcodificación) en tiempo real para atender a las demandas de los agentes de usuario y no tienen por qué servir un documento con una codificación de caracteres que cubra el conjunto completo de caracteres del documento.

Las codificaciones de caracteres usadas normalmente en la Web incluyen:

- ISO-8859-1 o "Latin-1" utilizable para la mayor parte de los idiomas de Europa occidental
- ISO-8859-5 (que soporta el alfabeto cirílico)
- SHIFT_JIS (una codificación japonesa)
- EUC-JP (otra codificación japonesa)
- UTF-8 (una codificación de ISO 10646 que utiliza un número diferente de bytes para distintos caracteres).

Especificación de la codificación de caracteres

Para determinar que codificación de caracteres se aplica a un documento, algunos **servidores** examinan los primeros bytes del documento, o comprueban una base de datos de ficheros conocidos y sus codificaciones. Los autores deberían enviar un parámetro "charset" siempre que sea posible, pero con cuidado de no identificar un documento con un valor de "charset" equivocado.

Para que un **agente de usuario** sepa qué codificación de caracteres ha sido utilizada, el servidor proporciona dicha información utilizando el parámetro "charset" del campo "Content-Type" de encabezado del protocolo http. Por ejemplo, el siguiente encabezado HTTP anuncia que la codificación de caracteres es EUC-JP:

```
Content-Type: text/html; charset=EUC-JP
```

El protocolo HTTP toma la ISO-8859-1 como la codificación de caracteres por defecto cuando el parámetro "charset" está ausente del campo de encabezado "Content-Type". Sin embargo esto es inútil ya que algunos servidores no permiten que se envíe un parámetro "charset" o no están configurados para enviar el parámetro.

Por este motivo, los documentos HTML pueden incluir información sobre la codificación de caracteres del documento a través del elemento **META**, de la siguiente manera:

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

Esta declaración **META** debe aparecer lo antes posible dentro del elemento **HEAD** y sólo debe utilizarse cuando en la codificación de caracteres los bytes con valores ASCII correspondan a caracteres ASCII (al menos hasta que se interprete el elemento **META**).

Cuando ni el protocolo HTTP ni el elemento **META** proporcionan información sobre la codificación de caracteres, el atributo **charset** puede aplicarse a otros elementos, asegurándose así que cuando el usuario reciba un recurso, el

agente de usuario reconocerá la codificación de caracteres.

Entonces, los agentes de usuario al determinar la codificación de caracteres de un documento deben tener en cuenta la siguiente prioridad (de más alta a más baja):

1. Un parámetro "charset" HTTP en un campo "Content-Type".
2. Una declaración **META** con **http-equiv** establecido en **Content-Type** y un valor dado para **charset**.
3. El atributo **charset** establecido en un elemento que designe un recurso externo.
4. Las preferencias del usuario.

REFERENCIAS DE CARACTERES

En ciertas ocasiones, los autores tendrán que hacer uso de las **referencias de caracteres**, especialmente cuando una codificación no puede expresar todos los caracteres del conjunto, o cuando las limitaciones de hardware y software no permiten hacerlo. **Las referencias de caracteres son un mecanismo independiente de la codificación de caracteres para introducir cualquier carácter del conjunto de caracteres del documento y pueden aparecer de dos maneras:**

- Referencias numéricas de caracteres (decimales o hexadecimales).
- Referencias a entidades de caracteres.

Referencias numéricas de caracteres

Las **referencias numéricas de caracteres** especifican la posición del código de un carácter en el conjunto de caracteres del documento, y pueden tener dos formas:

- La sintaxis "&#D;", se refiere al carácter de ISO 10646 con el número decimal **D**.
- La sintaxis "&#xH;" o "&#XH;", se refiere al carácter de ISO 10646 con el número hexadecimal **H**.

Algunos ejemplos son:

- **å**; (en decimal) representa la letra "a" con un circulito encima (usada, por ejemplo, en noruego).
- **å**; (en hexadecimal) representa el mismo carácter.
- **И**; (en decimal) representa la letra mayúscula cirílica "I".
- **水**; (en hexadecimal) representa el carácter chino para "agua".

Referencias a entidades de caracteres

Las **referencias a entidades de caracteres**, son una manera más intuitiva de referirse a caracteres del conjunto de caracteres del documento. Éstas utilizan nombres simbólicos para que los autores no tengan que recordar posiciones de código. Por ejemplo, la referencia a la entidad de caracteres **å** se refiere a la letra minúscula "a" con un anillo ("ring") encima, es más fácil de recordar que **å**; y si distinguen entre mayúsculas y minúsculas.

No se define una referencia a entidades de caracteres para cada carácter del conjunto de caracteres del documento.

Hay cuatro referencias a entidades de caracteres que se usan frecuentemente:

- **<**; representa el signo **<**, y debe usarse en su reemplazo para evitar la confusión con el comienzo de una etiqueta.
- **>**; representa el signo **>** y debe usarse en su reemplazo para que no sea confundido como el cierre de una etiqueta.
- **&**; representa el signo **&**, y debe usarse para no confundirlo con el comienzo de una referencia de

caracteres. Puede utilizar dentro de los valores de atributos, por ejemplo, en un URI.

- `"`; representa el signo " , y puede utilizarse para reemplazar las comillas, ya que estas se usan para delimitar los valores de atributos.

CARACTERES NO REPRESENTABLES

Un agente de usuario puede no ser capaz de representar correctamente todos los caracteres de un documento, porque carece de una fuente apropiada o porque un carácter tiene un valor que no puede expresarse con la codificación de caracteres interna del agente de usuario. En esos casos los agentes de usuario podrían:

- Adoptar un mecanismo para alertar al usuario sobre los recursos ausentes.
- Si los caracteres no disponibles se presentan usando su representación numérica, usar la forma hexadecimal (no la forma decimal), ya que ésta es la forma utilizada en los estándares de conjuntos de caracteres.

2. Interpretar SGML y el DTD



INTRODUCCIÓN AL SGML

Tal como dijimos, SGML es un lenguaje para la descripción de lenguajes de etiquetado que dan formato a documentos (markup languages).

Permite a los autores representar información estructural, presentacional y semántica junto con el contenido. El XHTML es un ejemplo de lenguaje de formato de documentos o aplicación SGML y junto con otras aplicaciones SGML se caracteriza por:

- Una **declaración SGML** que especifica qué caracteres y delimitadores pueden aparecer en la aplicación (ver CD > Anexos > xhtml1.dcl).
- Una **definición del tipo de documento** (DTD) que define la sintaxis de las estructuras de formato, junto a definiciones adicionales, tales como **referencias a entidades de caracteres**. Cada documento contiene una referencia al DTD que debe usarse para interpretarlo. (ver CD > Anexos > xhtml1-strict-1.dtd | xhtml1-transitional.dtd | xhtml1-frameset.dtd | xhtml-lat1.ent | xhtml-special.ent | xhtml-symbol.ent)
- Una **especificación** que describe la semántica que se debe conferir al código de formato. Esta especificación también impone restricciones de sintaxis que no pueden expresarse dentro del DTD (ver <http://www.w3.org/TR/xhtml1>).

Nota: En la Estos archivos deben abrirse en el block de notas de Windows para visualizarse correctamente.

ESTRUCTURAS SGML USADAS EN XHTML

Las secciones siguientes presentan las estructuras de SGML que se usan en XHTML.

Elementos

Un DTD SGML declara *tipos de elementos* que representan estructuras o un comportamiento deseado. HTML incluye elementos que representan párrafos, vínculos de hipertexto, listas, tablas, imágenes, etc.

Cada declaración de tipo de elemento describe generalmente tres partes: una etiqueta inicial, un contenido y una etiqueta final.

El nombre del elemento aparece en la **etiqueta inicial** (escrita `<p>`) y en la **etiqueta final** (escrita `</p>` - observe la / que precede al nombre del elemento). Por ejemplo, la etiqueta inicial y final del tipo de elemento **p** delimitan un párrafo:

```
<p>Un párrafo escrito con un tipo de elemento XHTML.</p>
```

En XHTML, a diferencia de HTML, no está permitido omitir las etiquetas finales o iniciales de los elementos, ya que todos deben tener su correspondiente apertura y cierre.

Algunos tipos de elemento no tienen contenido, son **elementos vacíos** que nunca tienen etiquetas finales. El DTD y la explicación de cada elemento indican si un elemento es vacío y si no lo es, cuál es el contenido legal. Por ejemplo, el elemento de salto de línea **BR** no tiene contenido; su único papel es terminar una línea de texto.

Nota: Los elementos no son etiquetas. Un *elemento* es la unidad estructural sintáctica del lenguaje del documento, mientras que la *etiqueta* es el delimitador de inicio y final del código de un elemento. Por ejemplo, en HTML el elemento **HEAD** está siempre presente, incluso si la etiqueta inicial y final de **HEAD** están ausentes.

Atributos

Los elementos pueden tener propiedades asociadas, llamadas **atributos**, que pueden tener valores (por defecto o asignados por el autor o por un script). Las parejas atributo/valor aparecen antes del cierre (`>`) de la etiqueta inicial de un elemento. En la etiqueta inicial de un elemento puede aparecer en cualquier orden cualquier número de parejas (legales) atributo/valor, separadas por espacios.

Como ejemplo se puede establecer el atributo **id** y el atributo **lang** de un encabezado:

```
<h1 id="importante" lang="es">Primer Título</h1>
```

Por defecto, SGML requiere que todos los valores de atributo estén delimitados por comillas dobles (`"` o `"`). Un valor de atributo delimitado por comillas dobles puede contener signos de comillas simples (`'`), y viceversa. El valor de los atributos sólo puede contener letras (a-z y A-Z), dígitos (0-9), guiones (ASCII decimal 45), puntos (ASCII decimal 46), sub-guiones (ASCII decimal 95) y dos puntos (ASCII decimal 58).

Referencias de caracteres

Las **referencias de caracteres** son nombres simbólicos o numéricos de caracteres que pueden incluirse en un documento y son útiles para hacer referencia a caracteres usados esporádicamente, o que son difíciles o imposibles de introducir con las herramientas de creación. Estas comienzan con un signo "&" y terminan con un punto y coma (;), tal como los siguientes ejemplos:

- `<` representa el signo `<`.
- `>` representa el signo `>`.
- `"` representa el signo `"`.

Comentarios

Los comentarios no tienen un significado especial, forman parte del código y tienen la sintaxis siguiente:

```
<!-- comentario de una línea -->  
<!-- un comentario,  
que ocupa más de una línea -->
```

No se permite espacios en blanco entre el delimitador de apertura de declaración de etiqueta (`<!`) y el delimitador de apertura de comentario (`--`), pero sí se permite entre el delimitador de cierre de comentario (`--`) y el delimitador de cierre de declaración de etiqueta (`>`). Se debe evitar colocar dos o más guiones (`---`) dentro de un comentario.

CÓMO LEER EL DTD DE HTML

El tutorial que sigue permitirá a los lectores que no estén familiarizados con SGML leer el DTD y entender los detalles técnicos de la especificación HTML.

Comentarios DTD

En los DTDs, los comentarios pueden extenderse a lo largo de una o más líneas, son únicamente informativos y están delimitados por un par de marcas `--`, ej.:

```
<!ELEMENT PARAM - O EMPTY -- valor de propiedad con nombre -->
```

Definiciones de entidades paramétricas

El DTD tiene una serie de definiciones de entidades paramétricas que definen una clase de macro que sólo puede aparecer en el DTD y a la que se puede hacer referencia para ser expandida en cualquier otra parte del DTD. Cuando se hace referencia a una entidad paramétrica por su nombre en el DTD, ésta se expande, sustituyéndose por una cadena.

Una definición de **entidad paramétrica** comienza con la palabra clave `<!ENTITY %` seguida por el nombre de la entidad, la cadena a la que se expande la entidad entre comillas, y por último un `>` de cierre.

```
<!ENTITY % fontstyle "TT | I | B | BIG | SMALL">
```

Luego, esa entidad podrá aparecer en el DTD iniciando con un `%`, seguido por el nombre de la entidad paramétrica, y finalmente por un `;` opcional.

```
% fontstyle;
```

Las cadenas pueden contener otros nombres de entidades paramétricas, que se pueden expandir recursivamente. Por ejemplo, la entidad `"%inline;"` se define de modo que incluya las entidades paramétricas `"%fontstyle;"`, `"%phrase;"`, `"%special;"` y `"%formctrl;"`.

```
<!ENTITY % inline "#PCDATA | %fontstyle; | %phrase; | %special; | %formctrl;">
```

Declaraciones de elementos

Básicamente el DTD declara **tipos de elementos** y sus **atributos**. La palabra clave `<!ELEMENT` comienza una declaración y el carácter `>` la termina. Entre ellos se especifica:

- El nombre del elemento.
- El contenido del elemento, si lo hay, que es denominado como su **modelo de contenido**. Los tipos de elemento que no tienen contenido se llaman elementos vacíos y son declarados con la palabra **"EMPTY"**.

EJEMPLO 1

```
<!ELEMENT ul (li)+>
```

- El tipo de elemento declarado es **UL** (`<!ELEMENT UL`)

- El modelo de contenido se ha declarado como "al menos un elemento LI". ((LI)+)

EJEMPLO 2 (ELEMENTO VACÍO)

```
<!ELEMENT img EMPTY>
```

- El tipo de elemento declarado es **IMG**.
- La palabra clave "EMPTY" significa que no debe tener contenido.

Definiciones del modelo de contenido

El **modelo de contenido** describe cuál puede ser el contenido de un tipo de elemento y puede incluir:

- Nombres de los tipos de elementos permitidos o prohibidos (Ej. el elemento **UL** contiene apariciones del tipo de elemento **LI**).
- Entidades DTD (Ej. **LABEL** contiene apariciones de la entidad paramétrica `%inline;`).
- Texto de documento (indicado mediante la estructura SGML "#PCDATA") que puede contener referencias de caracteres.

El modelo de contenido de un elemento se especifica de acuerdo a la sintaxis siguiente:

- **(...)** Delimita un grupo. La evaluación de cualquier sub-expresión dentro de paréntesis se realiza antes que lo que está fuera del mismo.
- **A** A debe aparecer una sola vez.
- **A+** Una o más instancias de A están son requeridas.
- **A?** Cero o una instancia de A están permitidas.
- **A*** Cero o más instancias de A están permitidas.
- **A | B** O bien A o bien B deben aparecer.
- **A , B** Tanto A como B deben aparecer, y en ese orden.

EJEMPLOS:

El elemento **UL** debe contener uno o más elementos **LI**:

```
<!ELEMENT ul (li)+>
```

El elemento **DL** debe contener uno o más elementos **DT** o **DD** en cualquier orden:

```
<!ELEMENT dl (dt|dd)+>
```

El elemento **OPTION** sólo puede contener texto y entidades, tales como `&`:

```
<!ELEMENT option (#PCDATA)>
```

Declaraciones de atributos

La palabra clave **<!ATTLIST** comienza la declaración de los atributos que puede tomar un elemento, seguida por el nombre del elemento, una lista de definiciones de atributos y un **>** de cierre:

- El nombre del atributo.
- El tipo del valor del atributo o un conjunto explícito de valores posibles (Ej. CDATA, NAME, ID) (En la [Tabla completa de elementos](#) el tipo de valor va dentro de paréntesis luego del nombre del atributo, ej. `id (ID)`).
- Si el valor por defecto del atributo es:
- **implícito** (palabra clave **#IMPLIED**), el valor por defecto debe ser proporcionado por el agente de usuario (en algunos casos heredándolo de elementos padre)
- **siempre requerido** (palabra clave **#REQUIRED**) (En la definición de elementos, el atributo requerido es seguido de un asterisco, ej. `alt* (Text)`).

- **fijo** e igual a un valor dado (palabra clave **#FIXED**), es explícitamente un valor por defecto para el atributo (En la definición de elementos, si el atributo tiene un valor fijo, el nombre del atributo es seguido por un = y valor va entre comillas, ej. `xml:space ("preserve")`).

Nota: En la definición de elementos, los valores legales de un atributo se listan entre comillas (""), separados por barras (|) y dentro de paréntesis, ej. `method ("get"*|"post")`.

EJEMPLOS

El atributo **rowspan** requiere valores de tipo **NUMBER**. El valor por defecto está dado explícitamente y es "1".

```
rowspan      NUMBER      1      -- número de filas abarcado por la celda --
```

El atributo opcional **id** requiere valores de tipo **ID**.

```
id           ID           #IMPLIED -- identificador único a nivel de documento --
```

Entidades DTD en definiciones de atributos

Las definiciones de atributos también pueden contener referencias a entidades, como se ve en el elemento **LINK** que comienza con la entidad paramétrica **%attrs**;

```
<!ELEMENT LINK EMPTY -- un vínculo independiente del medio -->
<!ATTLIST LINK
  %attrs; -- %coreattrs, %il8n, %events --
  charset %Charset; #IMPLIED -- codificación de caracteres del recurso --
  href %URI; #IMPLIED -- URI del recurso vinculado --
  hreflang %LanguageCode; #IMPLIED -- código de idioma --
  type %ContentType; #IMPLIED -- tipo consultivo de contenido --
  rel %LinkTypes; #IMPLIED -- tipos de vínculos directos --
  rev %LinkTypes; #IMPLIED -- tipos de vínculos inversos --
  media %MediaDesc; #IMPLIED -- para representar en estos medios --
>
```

La entidad paramétrica **%attrs**; se ha establecido porque estos atributos se definen para la mayoría de los tipos de elementos de XHTML de la siguiente forma:

```
<!ENTITY % attrs "%coreattrs; %il8n; %events;";>
```

La entidad paramétrica **%coreattrs**; de la definición de **%attrs**; se expande como sigue:

```
<!ENTITY % coreattrs
  "id ID #IMPLIED
  class CDATA #IMPLIED
  style %StyleSheet; #IMPLIED
  title %Text; #IMPLIED"
>
```

Atributos booleanos

Algunos atributos actúan como variables booleanas, es decir que su aparición en la etiqueta inicial de un elemento implica que el valor del atributo es "verdadero" y su ausencia implica un valor "falso".

Los atributos booleanos sólo pueden tomar un valor legal: el propio nombre del atributo (Ej. `selected="selected"`). Por ejemplo:

```
selected (selected) #IMPLIED -- opción preseleccionada --
```

El atributo se iguala a "verdadero" si aparece en la etiqueta inicial del elemento:

```
<option selected="selected">
  ...contenidos...
</option>
```

3. Datos básicos



TIPOS DE DATOS PARA ELEMENTOS O ATRIBUTOS

Existen ciertos tipos de datos que pueden aparecer como contenido de un elemento o valor de un atributo en el DTD XMHTML.

Tipos de contenido

El modelo de contenido requiere que en algunos casos se permita texto como contenido de uno o más elementos, y para ello se utiliza el símbolo **PCDATA** (processed character data) que hace referencia a datos de carácter procesados. Un tipo de contenido puede ser definido como **EMPTY** (o vacío), lo que significa que el elemento no tiene contenido y su modelo de contenido es mínimo.

Tipos de atributos

En algunas ocasiones es necesario definir los tipos de valores de atributos o especificar el conjunto de valores permitidos para los atributos. Los siguientes tipos de atributos son utilizados en las definiciones:

Tipo de atributo	Definición
CDATA	Es una secuencia de caracteres tomados del conjunto de caracteres del documento y puede incluir entidades de caracteres, sin embargo, los autores no deberían declarar valores con espacios en blanco al principio o al final.
ID	Un identificador único de documento. Debe comenzar con una letra (A-Z, a-z) que puede estar seguida por un número cualquiera de letras, dígitos (0-9), guiones ("-"), sub-guiones ("_"), dos puntos (":") y puntos (".").
IDREF	Una referencia a un identificador único de documento, es decir al valor de un ID definido en otro atributo.

IDREFS	Una lista de identificadores únicos de documento, separados por espacios.
NAME	Un nombre con las mismas limitaciones de caracteres que ID.
NMTOKEN	Un nombre compuesto sólo por letras, dígitos, punto (.), guión (-), subrayado (_) y dos puntos (:)
NMTOKENS	Uno o más valores NMTOKEN separados por espacio en blanco.




















































Tipos de información para atributos

Junto con estos tipos de información predefinidos, XHTML define los siguientes tipos de información y su semántica:

Tipo de Información	Tipo de Atributo	Descripción
Character	CDATA	%Character; indica un carácter individual del conjunto de caracteres del documento de [ISO10646]. Estos caracteres individuales pueden especificarse mediante referencias de caracteres (Ej. <code>&amp; ;</code>).
Charset	CDATA	%Charset; se refiere a una codificación de caracteres cuyo valor debe ser una cadena (Ej. "euc-jp") del registro IANA.
Charsets	CDATA	Una lista separada por comas de tipos de codificación de caracteres (en base a Charset)
ContentType	CDATA	%ContentType; se refiere a los tipos MIME o tipos de contenido, formados por el nombre del tipo de contenido y del subtipo. Ej. "text/html", "image/png", "image/gif", "video/mpeg", "text/css", y "audio/basic".
ContentTypes	CDATA	Una lista separada por comas de tipos de contenido (en base a ContentType)
Coords	CDATA	Una lista de longitudes separada por comas usada en áreas definidas para establecer coordenadas.
Datetime	CDATA	<p>%Datetime; representa una cadena fecha/hora, y su formato es el siguiente y debe ser respetado con la misma puntuación:</p> <pre>AAAA-MM-DDThh:mm:ssDZH</pre> <p>que se traduce en:</p> <pre>2007-01-31T22:58:21DZH</pre> <p>El significado es el siguiente:</p> <p>AAAA = año con cuatro dígitos = 2007 MM = mes con dos dígitos = 01 (enero) DD = día del mes con dos dígitos (de 01 a 31) = 31 T = indica el comienzo de la hora, debe ser mayúscula hh = hora con dos dígitos (de 00 a 23) = 22 mm = minuto con dos dígitos (00 a 59) = 58 ss = segundo con dos dígitos (00 a 59) = 21 DZH = designador de zona horaria Z = indica UTC (Coordinated Universal Time) y debe ser mayúscula. +hh:mm = hora local que está hh:mm por delante del UTC. -hh:mm = hora local que está hh:mm por detrás del UTC.</p> <p>Para los segundos, minutos y horas, se puede utilizar el valor "00" en caso de no conocerse con precisión el valor real.</p>
LanguageCode	NMTOKEN	%LanguageCode; dentro del valor de un atributo se refiere a un código de idioma (no permite espacios en blanco).
Length(*)	CDATA	%Length; se refiere a una longitud y puede ser un %Pixel; o un porcentaje del espacio horizontal o vertical disponible. Así, el valor "50%" significa la mitad del espacio disponible.

Tipo de Información	Tipo de Atributo	Descripción
LinkTypes	CDATA	<p>%LinkTypes; se refiere a una lista de tipos de vínculos separados por espacios (que no permite caracteres de espacio en blanco). Estos valores no diferencian entre mayúsculas y minúsculas, por lo cuál "Alternate" y "alternate" significan lo mismo.</p> <ul style="list-style-type: none">• Alternate: Designa una versión alternativa del documento. Cuando se usa con el atributo hreflang, implica una versión traducida del documento y con el atributo media, implica una versión diseñada para un medio (o medios) diferentes.• Stylesheet: Se refiere a una hoja de estilo externa. Se usa junto al tipo de vínculo "Alternate" para ofrecer hojas de estilo alternativas seleccionables por el usuario.• Start: Se refiere al primer documento de un conjunto de documentos, diciendo a los motores de búsqueda qué documento es considerado como el punto de inicio.• Next: Se refiere al siguiente documento en una secuencia lineal de documentos. Los agentes de usuario pueden optar por precargar el documento marcado como "next".• Prev: Se refiere al documento anterior en una serie ordenada de documentos. Algunos agentes de usuario también soportan el sinónimo "Previous".• Contents: Se refiere a un documento que sirve como tabla de contenidos. Algunos agentes de usuario también soportan el sinónimo ToC (de "Table of Contents").• Index: Se refiere a un documento que proporciona un índice para el documento actual.• Glossary: Se refiere a un documento que proporciona un glosario de términos que pertenecen al documento actual.• Copyright: Se refiere al aviso de copyright del documento actual.• Chapter: Se refiere a un documento que actúa como capítulo en una colección de documentos.• Section: Se refiere a un documento que actúa como sección en una colección de documentos.• Subsection: Se refiere a un documento que actúa como subsección en una colección de documentos.• Appendix: Se refiere a un documento que actúa como apéndice en una colección de documentos.• Help: Se refiere a un documento que ofrece ayuda (más información, vínculos a otros recursos informativos, etc.)• Bookmark: Se refiere a una señal de lectura. Una señal de lectura (bookmark) es un vínculo a un punto de entrada importante dentro de un documento extenso.

Tipo de Información	Tipo de Atributo	Descripción
MediaDesc	CDATA	<p>%MediaDesc; es una lista separada por comas de los descriptores de medios, que permite a los agentes de usuario cargar y aplicar las hojas de estilo de manera selectiva. La lista de los descriptores de medios reconocidos es:</p> <ul style="list-style-type: none"> • screen: Para pantallas no paginadas de computadora. • tty: Para medios que utilicen una cuadrícula de caracteres de ancho fijo, como teletipos, terminales y dispositivos portátiles con posibilidades limitadas de representación. • tv: Para dispositivos tipo televisión (baja resolución, en color, desplazamiento limitado). • projection: Para proyectores. • handheld: Para dispositivos de mano (pantalla pequeña, monocromos, gráficos por mapas de bits, ancho de banda limitado). • print: Para material paginado, opaco, y para documentos que se ven en una pantalla en modo de presentación preliminar a la impresión. • braille: Para dispositivos táctiles braille. • aural: Para sintetizadores de voz. • all: Apropiado para todos los dispositivos. <p>El control de medios es útil cuando se aplica a hojas de estilo externas, ya que los agentes de usuario pueden obtener de la red únicamente aquellas hojas de estilo que se apliquen al dispositivo actual. Por ejemplo, los navegadores basados en voz pueden evitar la descarga de hojas de estilo diseñadas para la representación visual.</p>
MultiLength(*)	CDATA	<p>%MultiLength; es una multilongitud que puede ser un %Length; o una longitud relativa, que adopta la forma "i*", donde "i" es un entero. Cuando los agentes de usuario reparten espacio entre los elementos, adjudican primero las longitudes en píxeles y en porcentajes, y luego dividen el espacio sobrante entre las longitudes relativas. Cada longitud relativa recibe una porción del espacio disponible que es proporcional al entero que precede al "*". El valor "*" es equivalente a "1*", así, si hay disponibles 60 píxeles de espacio después de haber adjudicado el espacio en píxeles y en porcentajes, y las longitudes relativas que deben asignarse son "1*", "2*" y "3*", se asignarán 10 píxeles al 1*, 20 píxeles al 2* y 30 píxeles al 3*.</p>
MultiLengths	CDATA	Una lista separada por comas de ítems del tipo Multilength.
Number	CDATA	Uno o más dígitos. Debe contener al menos un dígito (0-9).
Pixels(*)	CDATA	Píxeles (%Pixels;): es un entero que representa un número de píxeles del papel o pantalla. Así, el valor "50" significa cincuenta píxeles.
Script	CDATA	<p>%Script; puede aparecer como contenido del elemento SCRIPT y como valor de los atributos de eventos intrínsecos. Los agentes de usuario no deben evaluar los datos de script como código HTML, sino que deben pasarlos tal y como están, como datos para un motor de scripts. La distinción entre mayúsculas y minúsculas depende del lenguaje de scripts utilizado.</p> <p>Los datos de scripts que estén contenidos en un elemento no pueden contener referencias de caracteres, pero los datos de scripts que sean el valor de un atributo sí pueden contenerlas.</p>

Tipo de Información	Tipo de Atributo	Descripción																		
Shape	rect circle poly default	La forma de una región cuyos posibles valores son: rect, circle, poly y default.																		
Stylesheet	CDATA	%StyleSheet; puede aparecer como contenido de un elemento STYLE y como valor de un atributo style . Los agentes de usuario no deben evaluar los datos de estilo como código HTML. Los datos de hojas de estilo que están contenidos en un elemento no pueden contener referencias de caracteres, pero los datos de hojas de estilo que son el valor de un atributo sí pueden contenerlas.																		
Text	CDATA	Información textual arbitraria que debe ser legible por humanos.																		
URI	CDATA	Referencia a un Identificador Universal de Recursos (URI)																		
URIs	CDATA	Una lista separada por espacio de URIs.																		
Usado en Frameset o Transitional																				
Color	CDATA	<p>%Color; se refiere a las definiciones de colores. Un valor de color puede ser o bien un número hexadecimal (anteponiendo un signo "#") o uno de los siguientes dieciséis nombres de colores (que no distinguen entre mayúsculas y minúsculas):</p> <table><tbody><tr><td> Black = "#000000" (Negro)</td><td> Green = "#008000" (Verde)</td></tr><tr><td> Silver = "#C0C0C0" (Plateado)</td><td> Lime = "#00FF00" (Verde lima)</td></tr><tr><td> Gray = "#808080" (Gris)</td><td> Olive = "#808000" (Verde oliva)</td></tr><tr><td> White = "#FFFFFF" (Blanco)</td><td> Yellow = "#FFFF00" (Amarillo)</td></tr><tr><td> Maroon = "#800000" (Marrón)</td><td> Navy = "#000080" (Azul marino)</td></tr><tr><td> Red = "#FF0000" (Rojo)</td><td> Blue = "#0000FF" (Azul)</td></tr><tr><td> Purple = "#800080" (Púrpura)</td><td> Teal = "#008080" (Azul verdoso)</td></tr><tr><td> Fuchsia = "#FF00FF" (Fucsia)</td><td> Aqua = "#00FFFF" (Celeste)</td></tr><tr><td> Orange = "#FFA500" (Naranja)</td><td></td></tr></tbody></table> <p>Así, los valores de color "#800080" y "Purple" se refieren ambos al color púrpura.</p>	 Black = "#000000" (Negro)	 Green = "#008000" (Verde)	 Silver = "#C0C0C0" (Plateado)	 Lime = "#00FF00" (Verde lima)	 Gray = "#808080" (Gris)	 Olive = "#808000" (Verde oliva)	 White = "#FFFFFF" (Blanco)	 Yellow = "#FFFF00" (Amarillo)	 Maroon = "#800000" (Marrón)	 Navy = "#000080" (Azul marino)	 Red = "#FF0000" (Rojo)	 Blue = "#0000FF" (Azul)	 Purple = "#800080" (Púrpura)	 Teal = "#008080" (Azul verdoso)	 Fuchsia = "#FF00FF" (Fucsia)	 Aqua = "#00FFFF" (Celeste)	 Orange = "#FFA500" (Naranja)	
 Black = "#000000" (Negro)	 Green = "#008000" (Verde)																			
 Silver = "#C0C0C0" (Plateado)	 Lime = "#00FF00" (Verde lima)																			
 Gray = "#808080" (Gris)	 Olive = "#808000" (Verde oliva)																			
 White = "#FFFFFF" (Blanco)	 Yellow = "#FFFF00" (Amarillo)																			
 Maroon = "#800000" (Marrón)	 Navy = "#000080" (Azul marino)																			
 Red = "#FF0000" (Rojo)	 Blue = "#0000FF" (Azul)																			
 Purple = "#800080" (Púrpura)	 Teal = "#008080" (Azul verdoso)																			
 Fuchsia = "#FF00FF" (Fucsia)	 Aqua = "#00FFFF" (Celeste)																			
 Orange = "#FFA500" (Naranja)																				
FrameTarget	NMTOKEN	Nombre de marco utilizado como destino para el resultado de ciertas acciones.																		
ImgAlign	(top middle bottom left right)	Alineación utilizada para los elementos OBJECT , APPLET , IMG , INPUT y IFRAME																		
TextAlign	(left center right justify)	Alineación del texto derecha, izquierda, centrada o justificada.																		
LAlign	(top bottom left right)	Alineación del elemento LEGEND a izquierda o derecha y arriba o abajo.																		
TAlign	(left center right)	Emplazamiento horizontal de una tabla en relación con el documento.																		

(*)HTML especifica tres tipos de valores de longitud para los atributos:

- **Píxeles** (%**Pixels**;))
- **Longitud** (%**Length**;))
- **Multilongitud** (%**MultiLength**;))

TABLA DE ELEMENTOS

Existen numerosos elementos y atributos que se usaron desde los inicios del HTML y que con el paso del tiempo y las modificaciones de las especificaciones hasta la llegada del XHTML han caído en desuso por la presencia de nuevas estructuras. Por este motivo, sólo analizaremos aquellos elementos y atributos válidos en XHTML Strict. Los elementos y atributos para los DTD transicional y frameset sólo serán mencionados.

El desarrollo de los elementos y atributos, distribuidos según temas relacionados, será tratado en su totalidad en base a las especificaciones de la W3C, especialmente la especificación de HTML 4.01 (<http://www.w3.org/TR/html4/index/attributes.html>) y las adaptaciones a XHTML 1.0 declaradas en la recomendación del 26 de Enero del 2000 (<http://www.w3.org/TR/xhtml1/>).

Veamos a continuación una tabla de los elementos disponibles para XHTML separados por categorías, con una breve descripción, la página en la cuál están desarrollados y el DTD al que pertenecen:

Nombre	Vacío	DTD	Descripción
Estructura de un documento			
DOCTYPE			declaración del tipo de documento
HTML			elemento raíz del documento
HEAD			cabecera del documento
TITLE			título del documento
META	Vacío		meta-información genérica
BODY			cuerpo del documento
DIV			contenedor genérico de idioma/estilo
SPAN			contenedor genérico de idioma/estilo
H1			encabezado de nivel 1
H2			encabezado de nivel 2
H3			encabezado de nivel 3
H4			encabezado de nivel 4
H5			encabezado de nivel 5
H6			encabezado de nivel 6
ADDRESS			información sobre el autor
Dirección del texto			
BDO			anular algoritmo BiDi I18N
Texto			
P			párrafo
BR	Vacío		salto de línea forzado
PRE			texto preformateado
INS			texto insertado
DEL			texto borrado

Nombre	Vacío	DTD	Descripción
SUB			subíndice
SUP			superíndice
EM			énfasis
STRONG			énfasis fuerte
DFN			definición
CODE			fragmento de código de computadora
SAMP			ejemplo de salida de programas, scripts, etc.
KBD			texto que debe introducir el usuario
VAR			variable o argumento de un programa
CITE			cita
ABBR			abreviatura (Ej. WWW, HTTP, etc.)
ACRONYM			abreviatura (Ej. WAC, radar, ovni)
BLOCKQUOTE			cita larga
Q			cita corta en línea
Listas			
OL			lista ordenada
UL			lista no ordenada
LI			objeto de lista
DL			lista de definiciones
DT			término definido
DD			descripción de una definición
Tablas			
TABLE			tabla
CAPTION			título de tabla
COLGROUP			grupo de columnas de una tabla
COL	Vacío		columna de una tabla
THEAD			cabecera de tabla
TBODY			cuerpo de tabla
TFOOT			pie de tabla
TR			fila de una tabla
TD			celda de datos de una tabla
TH			celda de encabezado de tabla
Vínculos			
A			origen o destino de vínculo
LINK	Vacío		un vínculo independiente del medio
BASE	Vacío		URI base del documento
Objetos, imágenes y aplicaciones			
IMG	Vacío		imagen incluida
OBJECT			objeto incluido genérico
PARAM	Vacío		valor de propiedad con nombre
MAP			mapa de imágenes en el lado del cliente
AREA	Vacío		área de un mapa de imágenes en el lado del cliente

Nombre	Vacío	DTD	Descripción
Estilos			
STYLE			información de estilo
Fuentes y separadores horizontales			
TT			estilo de texto de teletipo o monoespacio
B			estilo de texto en negrita
I			estilo de texto en itálica
BIG			estilo de texto grande
SMALL			estilo de texto pequeño
HR	Vacío		separador horizontal
Marcos			
FRAMESET		DTD p/ marcos	subdivisión en ventanas
FRAME	Vacío	DTD p/ marcos	subventana
NOFRAMES		DTD p/ marcos	contenedor de contenido alternativo para la representación no basada en marcos
IFRAME		DTD p/ transicional	subventana en línea
Formularios			
FORM			formulario interactivo
INPUT	Vacío		control de formulario
BUTTON			botón
SELECT			selector de opciones
OPTGROUP			grupo de opciones
OPTION			opción seleccionable
TEXTAREA			campo de texto multilínea
LABEL			texto del rótulo de un campo de formulario
FIELDSET			grupo de controles de un formulario
LEGEND			leyenda de un grupo de campos
Scripts			
SCRIPT			sentencias de script
NOSCRIPT			contenedor de contenido alternativo para la representación no basada en scripts

Nota: Ver [tabla completa con los elementos XHTML](#), sus correspondientes atributos y el modelo de contenido.

Elementos en bloque y elementos en línea

De la lista de elementos antes nombrados, existen los denominados:

- **elementos "en bloque"** (o también "a nivel de bloque") o
- **elementos "en línea"** (o "a nivel de texto").

		Nombre de categoría	Elementos abarcados
ELEMENTOS EN FLUJO	Elementos en Línea - %Inline (cubre elementos inline o a nivel de texto)	#PCDATA	
		%inline;	%special %special.pre; (br span bdo map) object img
			%fontstyle tt i b big small
			%phrase em strong dfn code q samp kbd var cite abbr acronym sub sup
			%inline.forms input select textarea label button
			a
		%misc.inline	ins del script
	Elementos en Bloque - %Block	%block;	%heading h1 h2 h3 h4 h5 h6
			%lists ul ol dl
			%blocktext pre hr blockquote address
			p div fieldset table
		%misc	%misc.inline; noscript
		form	

Luego de este cuadro, podemos resumir que los elementos:

- **Inline** son los de las categorías: #PCDATA | %inline; | %misc.inline; , es decir:
#PCDATA | br | span | bdo | map | object | img | tt | i | b | big | small | em | strong | dfn | code | q | samp | kbd | var | cite | abbr | acronym | sub | sup | input | select | textarea | label | button | a | ins | del | script
- **Block** son los de las categorías: %block; | form | %misc;
h1|h2|h3|h4|h5|h6 | ul | ol | dl | pre | hr | blockquote | address | p | div | fieldset | table | ins | del | script | noscript | form
- **Flow** son los de Inline y Block.

Esta distinción se basa en:

Modelo de contenido

- Los elementos en bloque pueden contener elementos en línea y a otros elementos en bloque.
- Los elementos en línea sólo pueden contener datos y a otros elementos en línea.
- Los elementos en bloque crean estructuras "más grandes" que los elementos en línea.

Formato

- Los elementos en bloque tienen por defecto un formato diferente que el de los elementos en línea y comienzan en una nueva línea, y los elementos en línea no.

Direccionalidad

- Los elementos en bloque y en línea difieren en el modo de heredar la información de direccionalidad.

ATRIBUTOS GENÉRICOS POR CATEGORÍAS

Hay ciertos atributos que son utilizados de forma recurrente en los elementos, y que han sido separados por categorías para un fácil reconocimiento:

Atributo	Tipo de atributo	Valor	Descripción
Atributos principales: coreattrs			
id	ID	#IMPLIED	Identificador único de documento
class	CDATA	#IMPLIED	Lista de clases separadas por espacio
style	%StyleSheet;	#IMPLIED	Información de estilo asociada en línea
title	%Text;	#IMPLIED	Título informativo para mostrar como tooltip
Atributos de lenguaje: i18n			
lang	%LanguageCode;	#IMPLIED	Establece el código de idioma
xml:lang	%LanguageCode;	#IMPLIED	Código de idioma (según especificación XML 1.0)
dir	(ltr rtl)	#IMPLIED	Establece la dirección del texto
Atributos de teclado			
accesskey	%Character;	#IMPLIED	Tecla de acceso para acceder a un elemento
tabindex	%Number;	#IMPLIED	Orden de tabulación de un elemento
Atributos para eventos: events			
de teclado (no válidos en base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style y title)			
onkeypress	%Script;	#IMPLIED	Una tecla es presionada y liberada
onkeydown	%Script;	#IMPLIED	Una tecla es presionada
onkeyup	%Script;	#IMPLIED	Una tecla es liberada
de mouse (no válidos en base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style y title)			
onclick	%Script;	#IMPLIED	Un elemento es clicado
ondblclick	%Script;	#IMPLIED	Un elemento recibe un doble clic
onmousedown	%Script;	#IMPLIED	Un elemento es presionado
onmouseup	%Script;	#IMPLIED	Un elemento es liberado
onmouseover	%Script;	#IMPLIED	El puntero se coloca sobre el elemento
onmousemove	%Script;	#IMPLIED	El puntero se mueve sobre el elemento
onmouseout	%Script;	#IMPLIED	El puntero se quita del elemento
de ventana (sólo válidos para BODY y FRAMESET)			
onload	%Script;	#IMPLIED	Cuando finaliza la carga de una ventana
onunload	%Script;	#IMPLIED	Cuando se elimina un documento de una ventana
de formularios (sólo válidos para elementos de formulario)			
onchange	%Script;	#IMPLIED	Cuando pierde el foco y el contenido fue cambiado
onsubmit	%Script;	#IMPLIED	Cuando se envía un formulario
onreset	%Script;	#IMPLIED	Cuando se reinicia un formulario
onselect	%Script;	#IMPLIED	Cuando se selecciona texto de un campo de texto
de foco (válidos para A, AREA, LABEL, INPUT, SELECT, TEXTAREA y BUTTON)			
onfocus	%Script;	#IMPLIED	El elemento recibe el foco
onblur	%Script;	#IMPLIED	El elemento pierde el foco

IMPORTANTE: Uniendo los atributos coreattrs, i18n y los eventos de teclado y mouse se obtiene un conjunto mayor utilizado regularmente en el DTD, denominado **attrs**.

Notas de otros atributos:

- **xml:space** (en varios elementos): El espacio en blanco se define como en XML. Todos los espacios son preservados lo que se declara con el valor "preserve" que en este caso da el mismo resultado que el valor "default".
- **xmlns** (en el elemento html): se define como "http://www.w3.org/1999/xhtml"

Atributos principales

Atributo id

Sintaxis: `id="mi_id"`

El atributo **id** asigna un **identificador único** (o nombre) a un elemento que debe ser único en un documento. Cada elemento sólo puede tener un **id** y por tratarse de un nombre único, el valor de **id** puede ser utilizado como objetivo de un vínculo, o para definir una regla de estilo. El valor de **id** debe comenzar con un carácter alfabético o un guión bajo, el resto puede ser cualquier carácter alfanumérico.

Este atributo tiene varios papeles en HTML:

- Como **selector** para las hojas de estilo.
- Como **vínculo** destino para vínculos de hipertexto.
- Como medio de hacer referencia a un elemento en particular desde un **script**.
- Como nombre de un elemento **object** declarado.
- Para procesos generales por parte de agentes de usuario (Ej. para identificar campos cuando se transfieren datos desde páginas HTML hasta una base de datos, para traducir documentos HTML a otros formatos, etc.).

APLICACIÓN DE ID

Por ejemplo, los siguientes párrafos se distinguen entre sí por el nombre asignado a través del **id**:

```
<p id="primerparrafo">Mi primer párrafo con nombre único.</p>
<p id="segundoparrafo">Mi segundo párrafo con nombre único.</p>
```

Nota: El atributo **id** comparte el mismo espacio de nombres que el atributo **name** cuando se usa para nombres de vínculos.

Atributo class

Sintaxis: `class="mi_clase"`

El atributo **class** asigna uno o más nombres de clase a un elemento, y ese nombre puede asignarse a cualquier número de elementos, pudiéndose decir que el elemento pertenece a esas clases. Los nombres de clase múltiples deben estar separados por caracteres de espacio en blanco.

El atributo **class** tiene varios papeles en HTML:

- Como selector para hojas de estilo (cuando se desea asignar información de estilo a un conjunto de elementos).
- Para procesos generales por parte de agentes de usuario.

APLICACIÓN DE CLASS

En un formulario se puede codificar los párrafos con los atributos **id** y **class** para diferenciarlos según el tipo de resultado:

```
<p id="m1" class="correcto">Los datos fueron enviados correctamente.</p>
<p id="m2" class="advertencia ">Ciertos campos se enviaron vacíos.</p>
<p id="m3" class="error">El campo email es incorrecto.</p>
```

A través de las reglas de estilo CSS los agentes de usuario visuales sabrían que deben representar los mensajes en verde, amarillo y rojo:

```
p.correcto { color: green }
p.advertencia { color: yellow }
p.error { color: red }
```


En este caso el atributo **id** puede utilizarse para la programación o para mejorar la presentación de los mensajes individuales.

Atributo style

Sintaxis: `style="nombre:valor"`

Este atributo especifica información de estilo para el elemento que lo contiene. Se comporta como una regla de estilo en línea y sólo afecta a una pequeña porción del documento. La sintaxis del valor del atributo **style** está determinada por el lenguaje de hojas de estilo por defecto. Para los estilos CSS en línea, las declaraciones de propiedades son de la forma "**nombre:valor**" y están separadas por un punto y coma.

APLICACIÓN DE STYLE

Este ejemplo CSS establece información sobre el color y el tamaño de la fuente del texto de un párrafo específico.

```
<p style="font-size: 11px; color: black">Párrafo con estilo en línea</p>
```

Para especificar estilos para más de un elemento, se recomienda usar el elemento **STYLE**, y para lograr una flexibilidad óptima se recomiendan los estilos en hojas de estilo externas.

Atributo title

Sintaxis: `title="texto"`

Este atributo ofrece información consultiva sobre el elemento para el cuál se establece. A diferencia del elemento **TITLE**, que proporciona información sobre un documento entero y que sólo puede aparecer una vez, el atributo **title** puede anotar cualquier número de elementos. En la definición de cada elemento se establece si soporta o no este atributo. El nombre asignado puede ser cualquier carácter o palabra e incluso entidades de caracteres (`
` y ``; generarían un quiebre de línea). Un mismo título puede ser único o repetido dentro de un mismo documento HTML.

El valor del atributo **title** pueden ser representado por:

- navegadores visuales como un "tooltip" (un mensaje corto que aparece cuando el mouse se detiene sobre un objeto).
- agentes de usuario de voz pronunciando la información del título en un contexto similar.

APLICACIÓN DE TITLE

Frente al atributo **title** asignado a un vínculo, los agentes de usuario (visuales y no visuales) pueden decir la naturaleza del recurso vinculado:

```
<p>Para más información al respecto visite <a href="http://www.clarin.com.ar" title="Sitio web diario digital Clarin.com.ar">Clarín</a>.</p>
```

Atributos de lenguaje

Existen dos atributos que afectan a la internacionalización del HTML:

- la especificación del idioma (el atributo **lang**)
- la dirección (el atributo **dir**) del texto de un documento.

Atributo lang

Sintaxis: `lang="en"`

El atributo **lang** especifica el idioma base del contenido de un elemento y de los valores de los atributos que se utilizará para mostrar el texto y los caracteres en un sitio web. Esto permite la internacionalización del HTML a un gran número de lenguajes.

El valor por defecto de este atributo es desconocido y su objetivo es permitir a los agentes de usuario representar el contenido según la práctica cultural aceptada para un idioma dado. Lo que no implica que los agentes de usuario no deban representar aquellos caracteres que no son típicos de un idioma de la mejor forma posible.

La información proporcionada por **lang** puede ser útil para ayudar:

- a los motores de búsqueda
- a los sintetizadores de voz
- al agente de usuario
- a elegir variaciones de un signo para tipografía de alta calidad
- a elegir un conjunto de caracteres para el entrecomillado de citas
- a hacer decisiones sobre separación de palabras, ligaduras, y espaciado
- a los verificadores de ortografía y gramática

El valor de **lang** es un código de idioma que identifica un lenguaje natural hablado, escrito o usado de cualquier modo para la comunicación de información entre personas (los lenguajes de ordenador no están incluidos). Estos códigos de idioma están formados por un código principal de dos letras y un subcódigo (posiblemente vacío) con el siguiente formato (el subcódigo está entre paréntesis porque puede no aparecer):

```
codigoprincipal(-subcódigo)
```

En base a esto, el código de idioma del inglés serían "en" y del inglés estadounidense: "en-US". Otros código de idioma (códigos de dos letras) son: **fr** (francés), **de** (alemán), **it** (italiano), **nl** (neerlandés), **el** (griego), **es** (español), **pt** (portugués), **ar** (árabe), **he** (hebreo), **ru** (ruso), **zh** (chino), **ja** (japonés), **hi** (hindi), **ur** (urdu), y **sa** (sánscrito).

Un elemento hereda la información sobre el código de idioma en el siguiente orden (de más alto a más bajo):

- El atributo **lang** establecido en el propio elemento.
- El atributo **lang** establecido en el elemento padre más cercano.
- El encabezado HTTP "Content-Language" (configurado en un servidor).
- Valores por defecto del agente de usuario y preferencias del usuario.

Hay que tener en cuenta que en las tablas el atributo **lang** puede heredarse de la primera celda de un tramo, en lugar del padre.

APLICACIÓN DE LANG

Por ejemplo, si apareciera el signo **&gamma** dentro de una cita en español, los agentes de usuario deberían representar correctamente la puntuación de la cita para el español, y aún así intentar representar **&gamma** de la mejor manera posible:

```
<p><q lang="es">La radiación gamma (&gamma;) es un tipo de radiación
electromagnética producida, entre otras cosas, por fenómenos astrofísicos de gran
violencia</q>, dijo él profesor.</p>
```

RESULTADO

La radiación gamma (γ) es un tipo de radiación electromagnética producida, entre otras cosas, por fenómenos astrofísicos de gran violencia, dijo él profesor.

[🔗 Ver ejemplo de LANG en una cita \(CD > ejemplos > xhtml > estructura > lang1.html\)](#)

APLICACIÓN DE LANG DE FORMA REPETIDA

En el siguiente ejemplo, el elemento **HTML** declara el idioma como **en-US**. Sin embargo, en el cuerpo se encuentra un párrafo en español (**<p lang="es">**) después del cuál el texto sigue siendo interpretado en

inglés. En otro párrafo más adelante se encuentra una cita en portugués (`<q lang="pt">`), luego de la cuál el texto continúa en inglés.

```
<!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="en-US" lang="en-US">
<head>
  <title>Aplicación del atributo lang en todo el texto</title>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
  <meta name="author" content="Maira Vykus"/>
</head>
<body>
  <p>The first paragraph in english.</p>
  <p lang="es">El segundo párrafo en español.</p>
  <p>The third paragraph in english, but with a quote in portuguese, where
the man says <q lang="pt">Eu preciso saber da sua vida</q>. So then, the text
continues in english.</p>
</body>
</html>
```

🔗 [Ver ejemplo de LANG en todo el documento \(CD > ejemplos > xhtml > estructura > lang2.html\)](#)

Atributo dir

Sintaxis: `dir="rtl"`

El atributo **dir** especifica la dirección de un texto que no tiene una direccionalidad heredada, ya sea del contenido de un elemento, del valor de los atributos del mismo o de una tabla. Los valores posibles son:

- **ltr**: Texto o tabla de izquierda a derecha (Left-to-right).
- **rtl**: Texto o tabla de derecha a izquierda (Right-to-left).

Si es utilizado como atributo del elemento **HTML**, se aplica al documento entero, en cambio al usarse en otros elementos, sólo se aplica al texto bajo la influencia de dicho elemento.

La especificación UNICODE asigna direccionalidad a los caracteres y define un algoritmo (Ver “Direccionalidad”) para determinar la direccionalidad correcta del texto. Si un documento no contiene un carácter de derecha a izquierda, no se requiere que el agente de usuario aplique el algoritmo bidireccional. Si un documento contiene caracteres de derecha a izquierda, y si el agente de usuario muestra estos caracteres, el agente de usuario debe usar el algoritmo bidireccional.

UNICODE especifica caracteres especiales para aclarar la dirección del texto, sin embargo HTML ofrece código que hace lo mismo: el atributo **dir** y el elemento **BDO**. Así, para un párrafo en hebreo, es más intuitivo escribir:

```
<p lang="he" dir="rtl">...texto hebreo...</p>
```

que la referencias en UNICODE:

```
&#x202B;&#x05F4;...una cita en hebreo...&#x05F4;&#x202C;
```

El atributo **lang** no puede ser utilizado por los agentes de usuario para determinar la direccionalidad del texto, ya que esto es realizado con el atributo **dir**, que se hereda y puede ser anulado con el atributo **bdo**.

Nota: Ver el Anexo > [Direccionalidad](#) para más información sobre el algoritmo bidireccional y herencia.

Atributos de teclado

En un documento XHTML, el usuario debe **dirigir el foco hacia un elemento** para que éste se active y realice sus funciones. Hay varias maneras de dirigir el foco hacia un elemento:

- Designar el elemento con un dispositivo apuntador.
- Navegar de un elemento a otro con el teclado a través de la tecla **TAB**. El **orden de tabulación** puede

especificarse con el atributo **tabindex**. Una vez seleccionado, el elemento puede activarse con alguna otra secuencia de teclas.

- Seleccionar un elemento por medio de una **tecla de acceso** asignada con el atributo **accesskey** (a veces llamada "acelerador de teclado").

Atributo tabindex

Sintaxis: `tabindex="5"`

Este atributo especifica la posición del elemento actual dentro del orden de tabulación del documento actual y debe ser un número entre 0 y 32767. El orden de tabulación (puede incluir elementos anidados en otros elementos) define el orden en que el foco se dirige hacia los elementos cuando se navega por medio del teclado.

La secuencia real de teclas que permiten la navegación con tabulador o la activación de los elementos depende de la configuración del agente de usuario (Ej. la tecla **TAB** se usa para la navegación y la tecla **INTRO** se usa para activar el elemento seleccionado).

Los agentes de usuario deberían navegar por los elementos a los que puede dirigirse el foco de acuerdo con las siguientes reglas:

1. Navegar por los elementos que soporten el atributo **tabindex** y tengan asignado para éste un valor positivo, desde el elemento con menor valor de **tabindex** hasta el elemento con el valor más alto. Los valores no necesitan ser secuenciales ni deben comenzar por un valor en particular. Si hay elementos con valores idénticos debe navegarse por ellos según el orden en que aparezcan en el flujo de caracteres.
2. A continuación navegar por aquellos elementos que no soporten el atributo **tabindex** o por los que soportándolo tengan asignado para él un valor "0". Se navega por estos elementos según el orden en que aparezcan en el flujo de caracteres.
3. Los elementos que estén deshabilitados no participan en el orden de tabulación.

Los siguientes elementos soportan el atributo **tabindex**: **A**, **AREA**, **BUTTON**, **INPUT**, **OBJECT**, **SELECT** y **TEXTAREA**.

APLICACIÓN DE TABINDEX

En este ejemplo, el orden de tabulación será:

- **BUTTON**
- **INPUT** en orden (si bien **BUTTON** y el primer **INPUT** comparten el **tabindex**, **BUTTON** aparece primero)
- el vínculo creado por el elemento **A**.

```
<p>Ir a <a href="http://www.clarin.com" tabindex="10" title="clarin.com">Clarín.</a>

<button type="button" name="obtener_datos" tabindex="1" onclick="obtener_datos">
Obtener los datos actuales.</button>

<form action="http://www.dominio.com/contratar method="post">
  <input tabindex="1" type="text" name="nombre" />
  <input tabindex="2" type="text" name="apellido" />
  <input tabindex="3" type="submit" name="enviar" />
</form>
```

Atributo accesskey

Sintaxis: `accesskey="s"`

Este atributo designa un carácter del teclado, que al ser presionado junto con la tecla **ALT** o **META**, lleva el foco hacia un elemento HTML. Cambiar el foco significa que el cursor se dirigirá a dicho element (un vínculo o un input por ejemplo). El carácter puede ser cualquier carácter individual del teclado, incluyendo los alfabetos en mayúsculas y minúsculas, números, símbolos y signos de puntuación. No se puede repetir un carácter, ya que cada **accesskey** debe identificar a un único elemento.

La acción que tiene lugar cuando el foco se dirige hacia un elemento depende del elemento. Por ejemplo, cuando un usuario activa un vínculo definido con **A**, el agente de usuario normalmente sigue el vínculo y cuando activa un campo de texto, éste permite la entrada de texto, etc.

Los siguientes elementos soportan el atributo **ACCESSKEY: A, AREA, BUTTON, INPUT, LABEL, LEGEND y TEXTAREA**.

APLICACIÓN DE ACCESSKEY

Este ejemplo asigna la tecla de acceso "N" a un rótulo asociado con un control **INPUT**. Al pulsar la tecla de acceso, el foco se dirige hacia el rótulo, el cual a su vez lo dirige al control asociado. El usuario puede entonces introducir texto en el área **INPUT**.

```
<form action="..." method="post">
  <label for="nombre" accesskey="u">Nombre (u)</label>
  <input type="text" name="nombre" id="nombre" />
</form>
```

La invocación de teclas de acceso depende del sistema subyacente. Por ejemplo, en máquinas que ejecuten MS Windows, normalmente hay que pulsar la tecla **ALT** además de la tecla de acceso. En sistemas Apple, normalmente hay que pulsar la tecla **COMMAND** además de la tecla de acceso.

La representación de teclas de acceso depende del agente de usuario. Recomendamos a los autores que incluyan la tecla de acceso en el texto del rótulo o dondequiera que se aplique la tecla de acceso. Los agentes de usuario deberían representar las teclas de acceso de tal modo que se enfatice su papel y se distinga de otros caracteres (Ej. subrayándola).

Eventos de teclado

Evento onkeypress

Sintaxis: onkeypress="acción"

El evento **onkeypress** ocurre cuando se pulsa y se suelta una tecla encima de un elemento. Este atributo puede utilizarse con la mayoría de los elementos.

Evento onkeydown

Sintaxis: onkeydown="acción"

El evento **onkeydown** ocurre cuando se pulsa una tecla encima de un elemento. Este atributo puede utilizarse con la mayoría de los elementos.

Evento onkeyup

Sintaxis: onkeyup="acción"

El evento **onkeyup** ocurre cuando una tecla se suelta encima de un elemento. Este atributo puede utilizarse con la mayoría de los elementos.

Eventos de mouse

Evento onclick

Sintaxis: onclick="acción"

El evento **onclick** ocurre cuando se hace clic con el dispositivo apuntador sobre un elemento. Este atributo puede utilizarse con la mayoría de los elementos. Al hacer un clic simple sobre un elemento, el código de script del evento onclick es ejecutado. El script también puede llamar funciones o subrutinas que contienen código que correrá cuando el clic simple ocurra.

Evento ondblclick

Sintaxis: ondblclick="action"

El evento **ondblclick** ocurre cuando se hace doble clic con el dispositivo apuntador sobre un elemento. Este atributo puede utilizarse con la mayoría de los elementos. Al hacer doble clic sobre un elemento, el código de script del evento ondblclick es ejecutado. El script también puede llamar funciones o subrutinas que contienen código que correrá cuando el doble clic ocurra.

Evento onmousedown

Sintaxis: onmousedown="action"

El evento **onmousedown** ocurre cuando el botón del dispositivo apuntador se pulsa cuando está encima de un elemento. Este atributo puede utilizarse con la mayoría de los elementos.

Evento onmouseup

Sintaxis: onmouseup="action"

El evento **onmouseup** ocurre cuando el botón del dispositivo apuntador se suelta cuando está encima de un elemento. Este atributo puede utilizarse con la mayoría de los elementos.

Evento onmouseover

Sintaxis: onmouseover="action"

El evento **onmouseover** ocurre cuando el dispositivo apuntador se sitúa sobre un elemento. Este atributo puede utilizarse con la mayoría de los elementos.

Evento onmousemove

Sintaxis: onmousemove="action"

El evento **onmousemove** ocurre cuando el dispositivo apuntador se mueve mientras está sobre un elemento. Este atributo puede utilizarse con la mayoría de los elementos.

Evento onmouseout

Sintaxis: onmouseout="action"

El evento **onmouseout** ocurre cuando el dispositivo apuntador se aparta de un elemento. Este atributo puede utilizarse con la mayoría de los elementos.

Eventos de ventana

Evento onload

Sintaxis: onload="action"

El evento **onload** ocurre cuando el agente de usuario finaliza la carga de una ventana o de todos los marcos de un **FRAMESET**. Cuando el evento ocurre, el código llama a una función que ejecuta una acción determinada. Este atributo puede utilizarse con los elementos **BODY** y **FRAMESET**.

Evento onunload

Sintaxis: onunload="action"

El evento **onunload** ocurre cuando el agente de usuario elimina un documento de una ventana o marco (también cuando se refresca o actualiza la página). Cuando el evento ocurre, el código llama a una función que ejecuta una acción determinada (por ejemplo, un mensaje de alerta con información importante para el usuario). Este atributo

puede utilizarse con los elementos **BODY** y **FRAMESET**.

Eventos de formulario

Evento onsubmit

Sintaxis: onsubmit="action"

El evento **onsubmit** ocurre cuando se envía un formulario. Sólo se aplica al elemento **form**.

Evento onreset

Sintaxis: onreset="action"

El evento **onreset** ocurre cuando se reinicializa un formulario. Sólo se aplica al elemento **form**.

Evento onselect

Sintaxis: onselect="action"

El evento **onselect** ocurre cuando un usuario selecciona texto de un campo de texto. Este atributo puede utilizarse con los elementos **input** y **textarea**.

Evento onchange

Sintaxis: onchange="action"

El evento **onchange** ocurre cuando un control pierde el foco de entrada y su valor ha sido modificado después de que el foco se dirigió hacia él. Este atributo se aplica a los siguientes elementos: **input**, **select** y **textarea**.

Eventos de foco

Evento onfocus

Sintaxis: onfocus="action"

El evento **onfocus** ocurre cuando el foco se dirige hacia un elemento, ya sea con el dispositivo apuntador o por navegación con tabulador. Este atributo puede utilizarse con los siguientes elementos: **A**, **AREA**, **LABEL**, **INPUT**, **SELECT**, **TEXTAREA** y **BUTTON**.

Evento onblur

Sintaxis: onblur="action"

El evento **onblur** ocurre cuando el elemento pierde el foco ya sea con el dispositivo apuntador o por navegación con tabulador. Puede utilizarse con los mismos elementos que onfocus.

4. Estructura de un documento



INTRODUCCIÓN A LA ESTRUCTURA DE UN DOCUMENTO HTML

Un documento se compone de varias partes, antes y después de las cuáles puede aparecer espacio en blanco (espacios, saltos de línea, tabulaciones y comentarios):

- una línea con información sobre la versión de HTML (DOCTYPE),
- el elemento HTML que contiene a :
 - HEAD: una sección de cabecera.
 - BODY o FRAMESET: un cuerpo, que contiene el contenido real del documento.

Un ejemplo de un documento HTML sencillo sería:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
  <head>
    <title>Documento HTML de prueba</title>
  </head>
  <body>
    <p>El primer párrafo.</p>
  </body>
</html>
```

[🔗 Ver ejemplo de archivo XHTML sencillo \(CD > ejemplos > xhtml > estructura > base_sencilla.html\)](#)

[🔗 Ver ejemplo de archivo XHTML con la cabecera completa \(CD > ejemplos > xhtml > estructura > base_completa.html\)](#)

Nota: Para copiar el código, una vez abierto el archivo en el navegador, haga clic derecho con el mouse sobre la página y seleccione la opción “Ver código fuente”.

ELEMENTO DOCTYPE (VERSIÓN DE XHTML)

Sintaxis:

```
<!DOCTYPE ...>
```

La etiqueta **DOCTYPE** se utiliza para declarar la definición del tipo de documento (DTD) en un documento XHTML.

XHTML es un subconjunto de SGML (Standardized Generalized Markup Language) y como tal, utiliza DTDs para definir el contexto del lenguaje. La W3C ha definido al DTD como "...una colección de declaraciones que, como colección, define la estructura legal, elementos y atributos que están disponibles para el uso en un documento que se ajusta al DTD."⁽²⁾

Entonces, el DTD XHTML define de forma precisa la gramática, reglas y sintaxis que debe aplicarse al código XHTML utilizado para **obtener un documento XHTML válido**.

Esta etiqueta es obligatoria, y debe aparecer en la primera línea de todo código XHTML, incluso antes del elemento **HTML**. En esta etiqueta debe respetarse el signo de admiración (!), las mayúsculas y minúsculas, la sintaxis y el hecho de que esta es la única etiqueta XHTML que no se cierra. Si la etiqueta **DOCTYPE** no está presente, entonces no se trata de código XHTML.

DTD

Existen tres tipos de DTDs que aplican al XHTML: Estricto (strict), Transicional (transitional) y para marcos (frameset). En las tres es importante obedecer a la sintaxis:

DTD XHTML 1.0 Estricto

Es la forma más estricta de DTD XHTML, y debería ser utilizado cuando se tiene la certeza que los usuarios tienen acceso a navegadores modernos que reconocen CSS, ya que este DTD se declara cuando se utilizan Hojas de Estilo en Cascada (CSS) para mejorar la apariencia y los estilos del documento.

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

DTD XHTML 1.0 Transicional

Este DTD se declara cuando se utiliza el mismo **HTML** para crear la apariencia de la web, en lugar de utilizar CSS.

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

DTD XHTML 1.0 para documentos con marco

Este DTD se declara cuando se particiona el documento HTML en dos o más marcos (usando o no CSS).

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

El URI que aparece en la declaración del tipo de documento permite a los agentes de usuario descargar el DTD y los conjuntos de entidades que sean necesarios. Los siguientes son los DTDs y conjuntos de entidades de XHTML 1.0:

- **DTD XHTML 1.0 por defecto estricto** (Ver: CD > Anexos > xhtml1-strict.dtd)
- **DTD XHTML 1.0 Transicional** (Ver: CD > Anexos > xhtml1-transitional.dtd)

2 W3C – Definición de DTD (<http://www.w3.org/TR/xhtml1/#general>)

- **DTD XHTML 1.0 con marcos** (Ver: CD > Anexos > xhtml1-frameset.dtd)
- **Caracteres de la ISO 8859-1 o Latin-1** (Ver: CD > Anexos > xhtml1-lat1.ent)
- **Símbolos, símbolos matemáticos y letras griegas** (Ver: CD > Anexos > xhtml1-symbol.ent)
- **Caracteres con significado en el código y de internacionalización** (Ver: CD > Anexos > xhtml1-special)

Las dos últimas letras de la declaración indican el idioma del DTD, que para HTML es siempre inglés ("EN").

Nota: Ver Anexo > [Información de referencia de SGML para XHTML 1.0](#) para información más detallada de SGML para XHTML.

ELEMENTO HTML

Sintaxis: (etiqueta inicial y final obligatoria)

```
<html>...</html>
```

El elemento **HTML** es el elemento raíz del documento, informa al navegador que se trata de un programa codificado como HTML y sólo puede ser precedido por la etiqueta **DOCTYPE** que indica la declaración del tipo de documento (DTD). Las etiquetas de inicio y cierre de este elemento fijan el inicio y final del documento y un documento XHTML bien formado debe contener etiquetas **HTML**, **HEAD**, **TITLE** y **BODY** correctamente anidadas y cerradas.

Atributos		Modelo de contenido	
Aceptados en XHTML strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id de lenguaje: lang, xml:lang, dir otros: xmlns (URI = "http://www.w3.org/1999/xhtml")		HEAD, BODY	

EJEMPLO

Así, un documento XHTML típico tiene esta estructura:

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

<html>

  <head></head>
  <body></body>

</html>
```

Se recomienda que la etiqueta de apertura del elemento **HTML** defina el idioma utilizado para el documento XHTML y para XML, con los atributos **lang** y **xml:lang** respectivamente. También debería contener la declaración para el nombre de espacio XML (**xmlns**):

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

ELEMENTO HEAD

Sintaxis: (etiqueta inicial y final obligatoria)

```
<head>...</head>
```

El elemento **HEAD** es el encabezado de un documento XHTML. Sirve como contenedor de otros elementos que controlan el contenido y la apariencia del cuerpo del documento:

- **TITLE**: define el nombre de un documento (debe aparecer una vez)
- **BASE**: define la URL por defecto (puede aparecer una vez o no estar)
- **SCRIPT**: define código de scripts (puede aparecer cero o más veces)
- **STYLE**: define reglas de hojas de estilo (puede aparecer cero o más veces)
- **META**: define palabras clave (puede aparecer cero o más veces)
- **LINK**: define un vínculo (puede aparecer cero o más veces)

Los agentes de usuario no representan los elementos que aparecen como contenido del **HEAD**, sin embargo, pueden poner dicha información a disposición de los usuarios a través de otros mecanismos. La etiqueta **HEAD** aparece inmediatamente después de la etiqueta **HTML** y antes de las etiquetas **BODY** o **FRAMESET**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id de lenguaje: lang, xml:lang, dir otros: profile (URIs)		TITLE BASE SCRIPT STYLE META LINK OBJECT)*	
Desarrollo de atributos			
profile profile="http://gmpg.org/xfn/11"	Este atributo es una lista de una o más direcciones URL (separadas por coma) en las que se puede encontrar uno o más perfiles de metadatos .		

EJEMPLO

Este ejemplo de encabezado hace referencia a un perfil que define propiedades útiles para indexar documentos. A las propiedades definidas en este perfil -- incluyendo "author", "copyright", "keywords" (palabras clave) y "date" (fecha) -- se les asignan valores mediante declaraciones **META** subsiguientes.

```
<head profile="http://www.vykus.com.ar/perfil/main">
<title>Título del documento</title>
<meta name="author" content="Maira Vykus" />
<meta name="copyright" content="XMundo Networks" />
<meta name="keywords" content="hosting, Argentina, servidores linux" />
<meta name="date" content="2007-11-06T08:49:37+00:00" />
</head>
```

Metadatos

Los **metadatos** son información sobre un documento más que contenido del propio documento y pueden ser especificados en HTML de diferentes maneras.

La especificación de metadatos implica dos pasos:

1. Declaración de una propiedad y de su valor. Puede hacerse de dos maneras:
 1. Desde dentro de un documento, por medio del elemento **META**.
 2. Desde fuera de un documento, vinculando los metadatos por medio del elemento **LINK**.
2. Referencia a un perfil en el que se definen la propiedad y sus valores legales, a través del atributo **profile** del elemento **HEAD**. Este perfil se aplica a todos los elementos **META** y **LINK** de la cabecera del documento.

ELEMENTO TITLE

Sintaxis: (etiqueta inicial y final obligatoria)

```
<title>...</title>
```

El elemento **TITLE** aparece dentro del elemento **HEAD** y es utilizado para proveer un nombre o título a un documento HTML. Sólo puede haber un elemento **TITLE** por documento, y aunque el mismo no se considera parte del flujo del texto, puede ser mostrado por los navegadores como el encabezado de la página o como el título de la ventana.

Si no se declara un título, los navegadores mostrarán uno por defecto. Sin embargo, como este elemento identifica y describe los contenidos de un documento, es conveniente proveer títulos descriptivos y con un significado claro y conciso. Por ejemplo, en lugar de usar un título como "Introducción", se debería escribir "Introducción a Internet y el hipertexto" que es más informativo.

Los títulos pueden contener texto (e incluso estar vacíos) y entidades de caracteres (para caracteres acentuados, caracteres especiales, etc.), pero no pueden contener código (etiquetas HTML o comentarios).

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id de lenguaje: lang, xml:lang, dir		PCDATA	

EJEMPLO

Aquí tenemos un ejemplo de título de documento:

```
<head>
  <title>Introducción a Internet y el hipertexto</title>
</head>
```

ELEMENTO META

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<meta />
```

El elemento **META** es utilizado para listar información del documento actual (Ej. el autor, la fecha de caducidad, una lista de palabras clave, etc.). Puede haber más de un elemento meta por página, los cuáles son ampliamente utilizados por los buscadores de Internet.

Cada elemento **META** especifica una pareja propiedad/valor, donde el atributo **name** identifica la propiedad y el atributo **content** especifica el valor de la propiedad (cuando el valor de **META** es un URI, se puede utilizar el elemento **LINK** en su lugar).

El elemento **META** puede utilizarse para especificar la información por defecto de un documento en los aspectos siguientes:

- El lenguaje por defecto de scripts.
- El lenguaje por defecto de hojas de estilo.
- La codificación de caracteres del documento.

También puede usarse para especificar palabras clave que ayuden a los motores de búsqueda a mejorar la calidad de los resultados de una búsqueda. Junto con el atributo lang, pueden filtrar los resultados de la búsqueda en relación con las preferencias de idioma del usuario: La efectividad de los motores de búsqueda puede incrementarse usando el elemento **LINK** para especificar vínculos a traducciones del documento en otros idiomas, a versiones en otros

medios (Ej. PDF), y a un punto de partida para examinar un grupo de documentos.

Si bien el elemento **META** es un mecanismo para la especificación de metadatos, hay ciertos elementos (**TITLE**, **ADDRESS**, **INS** y **DEL**) y atributos (**title** y **cite**) que pueden ser utilizados por los autores en lugar de **META** para especificar metadatos.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id de lenguaje: lang, xml:lang, dir otros: <u>name</u> (CDATA), <u>content</u> * (CDATA), <u>scheme</u> (CDATA), <u>http-equiv</u> (CDATA)		Empty	
Desarrollo de atributos			
name <i>name="cadena detexto"</i>	Identifica un nombre de propiedad. No existen valores definidos para este atributo, y se puede colocar cualquier texto.		
content <i>content="cadena detexto"</i>	Es un atributo requerido que especifica el valor de una propiedad. No existen valores definidos para este atributo, y se puede colocar cualquier texto.		
scheme <i>scheme="cadena detexto"</i>	Especifica un esquema o procedimiento que se usará para interpretar el valor de la propiedad. Esto debe ser coincidente con los perfiles listados con el atributo profile en el elemento HEAD . Los valores del atributo scheme dependen de la propiedad name y del profile asociado.		
http-equiv <i>http-equiv="content-type"</i> <i>http-equiv="expires"</i> <i>http-equiv="refresh"</i> <i>http-equiv="set-cookie"</i>	Puede utilizarse en lugar del atributo name para crear una dupla http-equiv/content , con lo cuál se simulan encabezados HTTP. Provee información usada para generar encabezados HTTP (sin embargo, la mayoría de los servidores proxy ignoran este atributo). Los valores permitidos son: content-type , expires , refresh y set-cookie .		

Ejemplos

AUTOR

En el siguiente ejemplo, el atributo meta indica el autor de un documento, especificando una **propiedad** (autor) y un **valor** a la misma (Maira Vykus):

```
<meta name="author" content="Maira Vykus" />
```

Se puede incluir el atributo **lang** para indicar el idioma del valor del atributo **content**, tal como el siguiente ejemplo, en el que se explica que el nombre del autor es en inglés:

```
<meta name="author" lang="en" content="George Bush" />
```

SCHEME

El atributo **scheme** proporciona a los agentes de usuario más contexto para la interpretación correcta de los metadatos. Como en el siguiente ejemplo que aclara el formato de fecha, como día-mes-año:

```
<meta name="date" scheme="DD-MM-AAAA" content="10-09-2007" />
```

En otras ocasiones, puede proporcionar información útil, aunque no crítica a los agentes de usuario, como aclarar que un número es un código ISBN:

```
<meta name="identificador" scheme="ISBN" content="0-8230-2355-9" />
```

HTTP-EQUIV

Una declaración como sigue, serviría para que los cachés determinen cuando deben obtener una nueva copia del documento:

```
<meta http-equiv="expires" content="Tue, 20 Aug 1996 14:25:27 GMT" />
```

Y la siguiente, sirve para actualizar la página cada 30 segundos.

```
<meta http-equiv="refresh" content="30" />
```

CODIFICACIÓN DE CARACTERES

Se puede especificar la codificación de caracteres de un documento como ISO-8859-5:

```
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-5" />
```

MOTORES DE BÚSQUEDA

Uso de **META** con el atributo **lang**, para que los buscadores puedan filtrar los resultados de la búsqueda en relación con las preferencias de idioma del usuario:

```
<!-- Para inglés americano -->
<meta name="keywords" lang="en-us"
      content="web hosting, Argentine, linux servers, web pages" />
<!-- Para español -->
<meta name="keywords" lang="es"
      content="alojamiento web, Argentina, servidores linux, páginas web" />
```

ELEMENTO BODY

Sintaxis: (etiqueta inicial y final obligatoria)

```
<body> . . . </body>
```

El elemento **BODY** se utiliza para indicar el inicio y final del contenido de un documento HTML, que puede ser presentado por un agente de usuario de distintas maneras:

- los navegadores visuales consideran el cuerpo como un lienzo sobre el que aparece el contenido: texto, imágenes, colores, gráficos, etc.
- los agentes de usuario por voz pronuncian el contenido.

Si se utilizan marcos, entonces el elemento **BODY** debe ser reemplazado por el elemento **FRAMESET**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de ventana: onload (Script), onunload (Script)		Block	

Ejemplos

Los elementos de presentación aparecían dentro del elemento **BODY**, sin embargo esa práctica fue desaprobada a favor de la especificación de la presentación de un documento dentro de las hojas de estilo. La aplicación de estas hojas de estilo se puede hacer de dos maneras:

ESTILOS INCRUSTADOS EN EL ENCABEZADO DEL DOCUMENTO:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba</title>
  <style type="text/css">
    body { background: white; color: black }
    p { color: blue }
  </style>
</head>
<body>
  <p>El primer párrafo.</p>
</body>
</html>
```

ESTILOS VINCULADOS CON HOJAS DE ESTILO EXTERNAS:

Esto permite cambiar la presentación sin retocar el código XHTML:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba</title>
  <link rel="stylesheet" type="text/css" href="css.css">
</head>
<body>
  <p>El primer párrafo.</p>
</body>
</html>
```

ELEMENTO DIV

Sintaxis: (etiqueta inicial y final obligatoria)

```
<div>...</div>
```

El elemento **DIV** se utiliza para designar una porción en bloque de un documento HTML, y para aplicar los atributos principales al contenido de ese elemento. El comportamiento del elemento **DIV** es **en bloque**, ya que crea un quiebre de línea (equivalente a un elemento **BR**) antes y después del mismo. Estos elementos pueden estar anidados y permiten un gran nivel de control y manipulación de cada bloque individual de la página web, debido a que se pueden aplicar los atributos **class** y **style** para aplicar efectos de Hojas de Estilo en Cascada, usar el atributo **lang** para un texto en idioma extranjero o asignar un **id** a cada bloque para referenciarlo en un vínculo.

En resumen, el elemento **DIV** (junto con **SPAN**) permite añadir estructura a los documentos, especificando si su contenido es en bloque, pero sin imponer ningún otro estilo de presentación al contenido, que sólo será dado con la asignación de los atributos **id** y **class**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout	align (TextAlign)	Flow (acepta elementos en línea y en bloque)	

Ejemplo

CÓDIGO

```
<div>aaaaaaaaa</div>
<div>bbbbbbbbb</div>
```

RESULTADO

Los agentes de usuario visuales colocan un salto de línea antes y después de los elementos **DIV**:

```
aaaaaaaaa
bbbbbbbbb
```

[🔗 Ver ejemplo de DIV \(CD > ejemplos > xhtml > estructura > div.html\)](#)

ELEMENTO SPAN

Sintaxis: (etiqueta inicial y final obligatoria)

```
<span> . . . </span>
```

El elemento **SPAN** se utiliza para designar una porción en línea de un documento HTML, y para aplicar los atributos principales al contenido de este elemento. El comportamiento del elemento **SPAN** es **en línea**, ya que los efectos ocurren en el flujo normal del texto e imágenes (sin insertar quiebres de línea o retornos de carro). Al igual que los **DIV**, las etiquetas **SPAN** pueden estar anidadas y también permiten un gran control y manipulación de partes especiales de una página web, por la aplicación de los atributos **class**, **style**, **lang** entre otros.

En resumen, el elemento **SPAN** (junto con **DIV**) permite añadir estructura a los documentos, especificando si su contenido es en línea, pero sin imponer ningún otro estilo de presentación al contenido, que sólo será dado con la asignación de los atributos **id** y **class**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta sólo elementos en línea)	

Ejemplo

CÓDIGO

```
<span>aaaaaaaaa</span>
<span>bbbbbbbbb</span>
```

RESULTADO

Los agentes de usuario visuales colocan los elementos **SPAN** sin saltos de línea, con lo que el ejemplo anterior con **SPAN** resultaría en:

```
aaaaaaaaa bbbbbbbbbb
```

[🔗 Ver ejemplo de SPAN \(CD > ejemplos > xhtml > estructura > span.html\)](#)

ELEMENTOS H1, H2, H3, H4, H5, H6 (ENCABEZADOS)

Sintaxis: (etiqueta inicial y final obligatoria)

```
<h1>...</h1>
<h2>...</h2>
<h3>...</h3>
<h4>...</h4>
<h5>...</h5>
<h6>...</h6>
```

Los elementos **H1**, **H2**, **H3**, **H4**, **H5**, **H6** son utilizados para crear encabezados de texto para un documento y con ellos se describe brevemente el tema de la sección que introducen. Los encabezados se muestran en seis diferentes tamaños, cuya apariencia depende del navegador, y que va desde el **H1** cuya letra es la de mayor tamaño hasta el **H6** de menor medida. En base a esto, cabe aclarar que este elemento sólo debe ser utilizado para crear encabezados y títulos, y no para establecer tamaños de fuentes, ya que esta tarea puede realizarse de otra manera a través de hojas de estilo. Este elemento inserta automáticamente un retorno de carro y avance de línea al cerrar la etiqueta.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout	align (TextAlign)	Inline (acepta elementos en línea)	

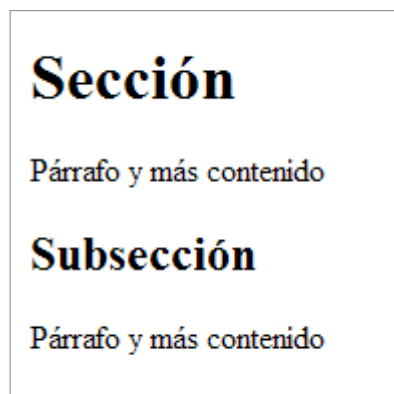
Ejemplo

CÓDIGO

```
<h1>Sección</h1>
<p>Párrafo y más contenido</p>
...
<h2>Subsección</h2>
<p>Párrafo y más contenido</p>
...
```

RESULTADO

Se puede generar un capítulo de un apunte del siguiente modo:



[Ver ejemplo de encabezados \(CD > ejemplos > xhtml > estructura > encabezados.html\)](#)

ELEMENTO ADDRESS

Sintaxis: (etiqueta inicial y final obligatoria)

```
<address>...</address>
```

El elemento **ADDRESS** indica que el texto es una dirección o información de contacto. Generalmente suele aparecer al principio del documento y puede contener cualquier etiqueta HTML. Los navegadores suelen representarlo en itálica, pero su apariencia puede ser personalizada a través de hojas de estilo.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplos

Información de contacto

CÓDIGO

Por ejemplo, una página podría incluir la siguiente información de contacto:

```
<address>
Contáctese con <a href="../staff/vykus/">Maira Vykus</a>,
webmaster de <a href="http://www.xmundo.net">XMundo.net</a>
</address>
```

RESULTADO

Contáctese con [Maira Vykus](#), webmaster de [XMundo.net](#)

Dirección postal

CÓDIGO

Una dirección postal la incluiría de la siguiente forma:

```
<address>
Universidad Empresarial Siglo 21.<br />
Av. Monseñor Pablo Cabrera Km. 8 ½.<br />
(5000) Córdoba, Argentina.
</address>
```

RESULTADO

*Universidad Empresarial Siglo 21.
Av. Monseñor Pablo Cabrera Km. 8 ½.
(5000) Córdoba, Argentina.*

🔗 [Ver ejemplos de ADDRESS \(CD > ejemplos > xhtml > estructura > address.html\)](#)

5. Texto



ELEMENTO P (PÁRRAFO)

Sintaxis: (etiqueta inicial y final obligatoria)

`<p>...</p>`

En XHTML el elemento **p** es el encargado de representar un **párrafo**. Un párrafo implica un quiebre de línea y un retorno de carro al mismo tiempo. La presentación visual de los párrafos se realiza básicamente a través de hojas de estilo, en aspectos tales como: tratamiento del espacio en blanco, saltos de línea y cambio automático de línea, justificación, división de palabras, formateo de los párrafos con respecto al contenido circundante, etc.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout	align (TextAlign)	Inline (acepta elementos en línea)	Elementos en bloque y otros p

Si bien la **representación visual** de los párrafos depende del agente de usuario, normalmente se muestran alineados a la izquierda y sin alinear por la derecha.

Las hojas de estilo proporcionan control sobre el tamaño y estilo de la fuente, los márgenes, los espacios antes y después de los párrafos, la sangría de la primera línea, la justificación, etc. Se podría redefinir los estilos para representar los párrafos sin los espacios que separan los párrafos, sin embargo, como esto puede confundir a los lectores, desaconsejamos esta práctica.

Ejemplo

CÓDIGO

```
Párrafo 1.
Párrafo 2.
Párrafo 3.

<p>Cuando yo tenía seis años vi en un libro sobre la selva virgen que se titulaba
"Historias vividas", una magnífica lámina. Representaba una serpiente boa que se
tragaba a una fiera.</p>

<p>En el libro se afirmaba: "La serpiente boa se traga su presa entera, sin
masticarla. Luego ya no puede moverse y duerme durante los seis meses que dura su
digestión".</p>

<p>Reflexioné mucho en ese momento sobre las aventuras de la jungla y a mi vez
logré trazar con un lápiz de colores mi primer dibujo.</p>
```

RESULTADO

Párrafo 1. Párrafo 2. Párrafo 3.

Quando yo tenía seis años vi en un libro sobre la selva virgen que se titulaba "Historias vividas", una magnífica lámina. Representaba una serpiente boa que se tragaba a una fiera.

En el libro se afirmaba: "La serpiente boa se traga su presa entera, sin masticarla. Luego ya no puede moverse y duerme durante los seis meses que dura su digestión".

Reflexioné mucho en ese momento sobre las aventuras de la jungla y a mi vez logré trazar con un lápiz de colores mi primer dibujo.

Nota: Al principio del código se han colocado los párrafos imaginarios 1, 2 y 3 sin la etiqueta `<p></p>` a su alrededor. Los agentes de usuario visuales muestran este texto uno al lado del otro, tal como se ve en el resultado. En cambio, cuando se envuelve un texto en la etiqueta `p`, este es renderizado con un quiebre de línea y un retorno de carro, que lo separa del siguiente párrafo existente.

[🔗 Ver ejemplo de párrafo \(CD > ejemplos > xhtml > texto > parrafo.html\)](#)

ELEMENTO BR (SALTO DE LÍNEA)

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<br />
```

El elemento **BR** finaliza o rompe forzosamente la línea actual de texto. Es análogo a un retorno de carro. Es una etiqueta de auto-cierre. El espacio anterior a la barra (/) es recomendado para una mayor compatibilidad con los agentes de usuario.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title	clear ("left" "all" "right" "none"*)	Empty	

Ejemplo

CÓDIGO

```
<p>Cuando yo tenía seis años vi en un libro sobre la selva virgen <br />que se titulaba "Historias vividas", una magnífica lámina. <br />Representaba una serpiente boa que se tragaba a una fiera.</p>
```

RESULTADO

Cuando yo tenía seis años vi en un libro sobre la selva virgen
que se titulaba "Historias vividas", una magnífica lámina.
Representaba una serpiente boa que se tragaba a una fiera.

Nota: En el ejemplo se verifica como el texto que sigue a la etiqueta **BR**, baja automáticamente al renglón siguiente, sin dejar un salto de línea adicional

🔗 [Ver ejemplos de salto de línea \(CD > ejemplos > xhtml > texto > br.html\)](#)

Entidad (impedir salto de línea)

En algunas ocasiones, se puede necesitar que dos o más palabras no sean separadas al llegar el final de un renglón. Para esta operación, entre las palabras en cuestión se debe colocar la entidad ** **; ante la cuál los agentes de usuario no deben asignar un salto de línea.

CÓDIGO

```
<p>Ejemplo 1: Cascade Stylesheet</p>  
<p>Ejemplo 2: Cascade&nbsp;Stylesheet</p>
```

RESULTADO

Ejemplo 1: Cascade
Stylesheet

Ejemplo 2:
Cascade Stylesheet

Nota: Si la palabra Cascade Stylesheet estuviera al final del renglón, en el ejemplo 1, Cascade quedaría en la primera línea y Stylesheet en el inicio de la segunda. Sin embargo, al colocar la entidad ** **; en el segundo ejemplo, ambas palabras serían enviadas juntas al segundo renglón, con lo cuál se anula el salto de línea entre dichas palabras.

🔗 [Ver ejemplo de impedimento de salto de línea \(CD > ejemplos > xhtml > texto > nobr.html\)](#)

ELEMENTO BLOCKQUOTE (CITA LARGA)

Sintaxis: (etiqueta inicial y final obligatoria)

```
<blockquote>...</blockquote>
```

El elemento **BLOCKQUOTE** es para citas largas con contenido en bloque. Es utilizado para separar un bloque de texto, una cita, del resto del documento. La apariencia depende del agente de usuario, sin embargo, generalmente está

sangrada de ambos lados y con saltos de línea antes y después.

Dentro de la misma se usan los elementos **P**, **BR** y otros para dar formato al texto. Se puede usar cualquier etiqueta, incluyendo **CITE** y **Q**.

Este elemento no debe ser utilizado para conseguir márgenes en un bloque de texto, ya que este mismo efecto puede obtenerse con hojas de estilo. Por este posible uso para asignar márgenes, los agentes de usuario no deberían agregar signos de puntuación de citas. En caso que el uso de **BLOCKQUOTE** se efectivamente una cita, el autor deberá usar hojas de estilo para agregar dichos signos antes y después de la misma.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout otros: cite (URI)		Block (acepta elementos en bloque)	
Desarrollo de atributos			
cite <i>cite="original.html"</i>	El valor de este atributo es un URI que designa un documento o mensaje original, que sirve para dar información sobre la fuente de la que se tomó la cita.		

Ejemplo

CÓDIGO

```
<p>Aquí viene la cita:</p>
<blockquote>
<p>Las personas mayores me aconsejaron abandonar el dibujo de serpientes boas,
ya fueran abiertas o cerradas, y poner más interés en la geografía, la historia,
el cálculo y la gramática. De esta manera a la edad de seis años abandoné una
magnífica carrera de pintor. Había quedado desilusionado por el fracaso de mis
dibujos número 1 y número 2. Las personas mayores nunca pueden comprender algo
por sí solas y es muy aburrido para los niños tener que darles una y otra vez
explicaciones.</p>
</blockquote>
```

RESULTADO

Aquí viene la cita:

Las personas mayores me aconsejaron abandonar el dibujo de serpientes boas, ya fueran abiertas o cerradas, y poner más interés en la geografía, la historia, el cálculo y la gramática. De esta manera a la edad de seis años abandoné una magnífica carrera de pintor. Había quedado desilusionado por el fracaso de mis dibujos número 1 y número 2. Las personas mayores nunca pueden comprender algo por sí solas y es muy aburrido para los niños tener que darles una y otra vez explicaciones.

[🔗 Ver ejemplo de blockquote \(CD > ejemplos > xhtml > texto > blockquote.html\)](#)

ELEMENTO Q (CITA CORTA)

Sintaxis: (etiqueta inicial y final obligatoria)

```
<q> . . . </q>
```

El elemento **q** sirve para citas cortas (contenido en línea) que no requiere cambios de párrafo. Esta etiqueta toma el texto seleccionado y lo muestra como una cita encerrada entre un par de dobles comillas. A diferencia del elemento **BLOCKQUOTE**, el **q** no es precedido o seguido por saltos de línea. Por lo tanto, la cita permanece en línea.

Los autores no deberían colocar marcas de citación al principio y al final del contenido de un elemento **q**, ya que los agentes de usuario visuales deben agregar las mismas. Estos tendrán que colocar las marcas de citación en función del idioma, respetando la representación de las citas externas y anidadas.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout otros: cite (URI)		Inline (acepta elementos en línea)	
Desarrollo de atributos			
cite cite="original.html"		El valor de este atributo es un URI que designa un documento o mensaje original, que sirve para dar información sobre la fuente de la que se tomó la cita.	

Ejemplo

Citas simples

CÓDIGO

```
<p>En el libro se afirmaba: <q>La serpiente boa se traga su presa entera, sin masticarla. Luego ya no puede moverse y duerme durante los seis meses que dura su digestión.</q></p>
```

RESULTADO

En el libro se afirmaba: “La serpiente boa se traga su presa entera, sin masticarla. Luego ya no puede moverse y duerme durante los seis meses que dura su digestión.”

Nota: Si bien las aplicaciones de usuario deberían incorporar las comillas, como Explorer no lo hace, este es un ejemplo tomado de Firefox.

[🔗 Ver ejemplo de Q \(CD > ejemplos > xhtml > texto > q.html\)](#)

Citas anidadas

CÓDIGO

```
<p>Estefanía dijo, <q>Mi padre decía, <q>Los libros son las abejas que llevan el polen de una inteligencia a otra.</q> Creo que escribía libros para sentirse parte de la polinización de las mentes.</q></p>
```

RESULTADO

Estefanía dijo, “Mi padre decía, ‘Los libros son las abejas que llevan el polen de una inteligencia a otra.’ Creo que escribía libros para sentirse parte de la polinización de las mentes.”

Nota: Tal como se ve en el ejemplo, la cita externa está rodeada por comillas dobles, y la interna por comillas simples.

[🔗 Ver ejemplo de Q anidados \(CD > ejemplos > xhtml > texto > q_anidados.html\)](#)

ELEMENTO EM

Sintaxis: (etiqueta inicial y final obligatoria)

```
<em> . . . </em>
```

El elemento **EM** indica **énfasis**. Si bien la presentación depende del agente de usuario, este elemento suele representarse con *itálica*. Los sintetizadores de voz pueden cambiar los parámetros de síntesis, como el volumen, el tono y la velocidad.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p>Para marcar un énfasis simple, <em>utilizo el elemento em</em>.</p>
```

RESULTADO

Para marcar un énfasis simple, *utilizo el elemento em.*

[🔗 Ver ejemplo de EM \(CD > ejemplos > xhtml > texto > em.html\)](#)

ELEMENTO STRONG

Sintaxis: (etiqueta inicial y final obligatoria)

```
<strong>...</strong>
```

El elemento **STRONG** indica un **énfasis más fuerte**. Si bien la presentación depende del agente de usuario, este elemento suele representarse con una fuente en **negrita**. Los sintetizadores de voz pueden cambiar los parámetros de síntesis, como el volumen, el tono y la velocidad.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p>Para marcar un énfasis fuerte, <strong>utilizo el elemento strong</strong>.</p>
```

RESULTADO

Para marcar un énfasis fuerte, **utilizo el elemento strong.**

[Ver ejemplo de STRONG \(CD > ejemplos > xhtml > texto > strong.html\)](#)

ELEMENTO CITE

Sintaxis: (etiqueta inicial y final obligatoria)

```
<cite>...</cite>
```

El elemento **CITE** contiene una cita o una referencia a otras fuentes. Generalmente se muestra en itálica y permanece en línea.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p>Como dijo <cite>Walt Disney</cite>, <q>Piensa, cree, sueña y atrévete.</q></p>
<p>Podrá encontrar más información en <cite>[ISO-9001]</cite>.</p>
```

RESULTADO

Como dijo *Walt Disney*, “Piensa, cree, sueña y atrévete.”

Podrá encontrar más información en *[ISO-9001]*.

[Ver ejemplo de CITE \(CD > ejemplos > xhtml > texto > cite.html\)](#)

ELEMENTO DFN

Sintaxis: (etiqueta inicial y final obligatoria)

```
<dfn>...</dfn>
```

El elemento **DFN** indica que aquí es donde se define el término encerrado. El mismo es mostrado en *itálica*. Es comúnmente utilizada en artículos de ingeniería, ciencia y técnica para marcar la primera aparición de un término.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p><dfn>HTML</dfn> es el acrónimo de Hypertext Markup Language.</p>
```

RESULTADO

HTML es el acrónimo de Hypertext Markup Language.

[Ver ejemplo de DFN \(CD > ejemplos > xhtml > texto > dfn.html\)](#)

ELEMENTO CODE

Sintaxis: (etiqueta inicial y final obligatoria)

```
<code>...</code>
```

El elemento **CODE** designa un fragmento de código de computadora en una fuente especial, generalmente una **mono espaciada**. Sin embargo, la apariencia depende del navegador y se deben utilizar otras etiquetas para lograr la apariencia deseada. El elemento **PRE** es el más usado para mostrar código.

Dentro del elemento **CODE**, no se pueden utilizar los delimitadores `<` y `>`, ya que estos son ejecutados por el navegador como código real. Los mismos deben ser sustituidos por `<` y `>`; respectivamente.

Generalmente, los elementos **CODE**, **KBD**, **SAMB** y **TT** tiene la misma apariencia visual en un mismo navegador.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<code>
<lt;body><br />
  <lt;p>ejemplo</p><br />
</body><br />
</code>
```

RESULTADO

```
<body>
<p>ejemplo</p>
</body>
```

Nota: Como se visualiza en el ejemplo, fue necesario colocar elementos **BR** para que cada una línea quede debajo de la otra. Por otra parte, este elemento no mantiene los espacios o tabulaciones que se colocan mediante el teclado.

[🔗 Ver ejemplo de CODE \(CD > ejemplos > xhtml > texto > code.html\)](#)

ELEMENTO SAMP

Sintaxis: (etiqueta inicial y final obligatoria)

```
<samp>...</samp>
```

El elemento **SAMP** designa una cadena corta de caracteres de la salida de un programa, script, etc. La apariencia depende del navegador, sin embargo suele usarse una tipografía mono espaciada o teletipo.

Generalmente, los elementos **CODE**, **KBD**, **SAMP** y **TT** tiene la misma apariencia visual en un mismo navegador.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p>Los elementos <samp>table</samp>, <samp>caption</samp>, <samp>tr</samp>, <samp>td</samp>, entre otros, son los utilizados para crear una tabla.</p>
```

RESULTADO

Los elementos table, caption, tr, td, entre otros, son los utilizados para crear una tabla.

[🔗 Ver ejemplo de SAMP \(CD > ejemplos > xhtml > texto > samp.html\)](#)

ELEMENTO KBD

Sintaxis: (etiqueta inicial y final obligatoria)

```
<kbd>...</kbd>
```

El elemento **KBD** indica texto que debe ser introducido por el usuario en su teclado. La apariencia depende del navegador, sin embargo suele usarse una tipografía mono espaciada o teletipo.

Por ejemplo, puede ser usado en manuales de programas, para indicarle al usuario que tecleando control + c puede copiar un texto seleccionado.

Generalmente, los elementos **CODE**, **KBD**, **SAMB** y **TT** tiene la misma apariencia visual en un mismo navegador.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p>Una vez seleccionado el texto en cuestión apriete las teclas <kbd>control</kbd>  
+ <kbd>C</kbd>.</p>
```

RESULTADO

Una vez seleccionado el texto en cuestión apriete las teclas control + C.

[🔗 Ver ejemplo de KBD \(CD > ejemplos > xhtml > texto > kbd.html\)](#)

ELEMENTO VAR

Sintaxis: (etiqueta inicial y final obligatoria)

```
<var>...</var>
```

El elemento **VAR** indica que el texto es una variable o un argumento de un programa. Se utiliza dentro de los elementos **CODE** y **PRE**, y generalmente es mostrado en itálica.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<pre>
if (document.getElementById) <var>blnDOM</var> = true;
</pre>
```

RESULTADO

```
if (document.getElementById) blnDOM = true;
```

[Ver ejemplo de VAR \(CD > ejemplos > xhtml > texto > var.html\)](#)

ELEMENTO ABBR

Sintaxis: (etiqueta inicial y final obligatoria)

<abbr>...</abbr>

Para la RAE, una *abreviación* es el "Procedimiento de reducción de una palabra mediante la supresión de determinadas letras o sílabas; p. ej., los acrónimos, los acortamientos, las abreviaturas y las siglas."⁽³⁾

El elemento **ABBR** indica una forma abreviada (Ej. WWW, HTTP, Sr., Dra., etc.). Implica que la palabra encerrada en las etiquetas es la abreviación de una palabra más larga o frase. Puede utilizarse el atributo title de este elemento para proporcionar la forma completa o expandida de la expresión.

Obsérvese que las abreviaturas y los acrónimos tienen a menudo pronunciaciones idiosincrásicas. Por ejemplo, mientras que "IRS" y "BBC" se suelen pronunciar letra por letra, "OTAN" y "UNESCO" se pronuncian fonéticamente. Y hay otras formas abreviadas (Ej. "URI" y "SCSI") que algunas personas deletrean y que otras pronuncian como palabras. Cuando sea necesario, los autores deberían usar hojas de estilo para especificar la pronunciación de una forma abreviada.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p><abbr title="World Wide Web">WWW</abbr> es una abreviación muy común en la web.</p>
```

RESULTADO

WWW es una abreviación muy común en la web.

3 Real Academia Española - Abreviación (http://buscon.rae.es/draeI/SrvltGUIBusUsual?TIPO_HTML=2&LEMA=abreviación)

Nota: Se recomienda colocar el significado de la abreviatura como valor del atributo **title**, lo cuál será mostrado como un tooltip por las aplicaciones de usuario.

🔗 [Ver ejemplo de ABBR](#) (CD > ejemplos > xhtml > texto > abbr.html)

ELEMENTO ACRONYM

Sintaxis: (etiqueta inicial y final obligatoria)

```
<acronym>...</acronym>
```

Según la Real Academia Española un *acrónimo* es:

m. Tipo de sigla que se pronuncia como una palabra; p. ej., o(bjeto) v(olante) n(o) i(dentificado).
m. Vocablo formado por la unión de elementos de dos o más palabras, constituido por el principio de la primera y el final de la última, p. ej., ofi(cina infor)mática, o, frecuentemente, por otras combinaciones, p. ej., so(und) n(avigation) a(nd) r(anging), Ban(co) es(pañol) (de) (crédi)to.⁽⁴⁾

El elemento **ACRONYM** indica un acrónimo (Ej. WAC, radar, ovni, sonar, Banesto etc.). Significa que el texto encerrado en dicho elemento es un tipo especial de abreviación formado usando la primera letra de cada frase. Puede utilizarse el atributo **title** de este elemento para proporcionar la forma completa o expandida de la expresión.

Al marcar estas estructuras se proporciona información útil a los agentes de usuario y a herramientas tales como correctores ortográficos, sintetizadores de voz, sistemas de traducción e indexadores de motores de búsqueda.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p><acronym>OVNI</acronym> significa Objeto Volador no Identificado.</p>
```

RESULTADO

OVNI significa Objeto Volador no Identificado.

🔗 [Ver ejemplo de ACRONYM](#) (CD > ejemplos > xhtml > texto > acronym.html)

ELEMENTO PRE

Sintaxis: (etiqueta inicial y final obligatoria)

```
<pre>...</pre>
```

4 Real Academia Española - Acrónimo (http://buscon.rae.es/draef/SrvltGUIBusUsual?TIPO_HTML=2&LEMA=acrónimo)

La etiqueta **PRE** es utilizada para mostrar texto preformateado. El resultado mostrará exactamente como el texto dentro del pre se encuentra renderizado, incluyendo los espacios en blanco, tabulaciones y saltos de línea. Esto permite mantener la apariencia de la información en filas y columnas, o con espacios especiales en caso de poesías, etc. El uso más común, tal como se muestra en este manual, es para mostrar código de computadora y el resultado correspondiente.

Los agentes de usuario visuales, normalmente muestran el texto en una fuente monoespaciada. Los elementos que no puede contener surgen por la intención de mantener constantes el espaciado de líneas y la alineación de columnas de texto representado con una fuente de ancho fijo. Se desaconseja a los autores alterar este comportamiento a través de hojas de estilo.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout xml:space ("preserve")	width (Number)	Inline (acepta elementos en línea)	IMG OBJECT BIG SMALL SUB SUP y elementos en Bloque

Ejemplo

CÓDIGO

```
<pre>
          CALORIAS      GRASA
Banana (100gr)    120 calorías    0,48 gr.
Manzana (100gr)  58 calorías     0 gr.
</pre>
```

RESULTADO

```

          CALORIAS      GRASA
Banana (100gr)    120 calorías    0,48 gr.
Manzana (100gr)  58 calorías     0 gr.
```

[Ver ejemplo de PRE \(CD > ejemplos > xhtml > texto > pre.html\)](#)

ELEMENTOS INS Y DEL

Sintaxis: (etiqueta inicial y final obligatoria)

```
<ins>...</ins>
<del>...</del>
```

Estos elementos son utilizados para marcar partes de un documento que han sido insertadas (**INS**) o borradas (**DEL**), en relación con una versión diferente del documento.

La etiqueta **DEL** marca el texto a ser borrado, cruzando una línea horizontal a través de los caracteres. Por el contrario, la etiqueta **INS**, designa el texto agregado con un subrayado. Esto puede ser modificado usando hojas de estilo.

Estos dos elementos pueden actuar como elementos en bloque o como elementos en línea (pero no como ambos a la vez). También pueden contener una o más palabras dentro de un párrafo o uno o más elementos en bloque como párrafos, listas y tablas. Una etiqueta **INS** no puede aparecer entre la apertura y cierre de una etiqueta **DEL** y

viceversa.

Atributos aceptados		Modelo de contenido	
XHTML Strict		Frameset,Transitional	Contiene a: No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout <u>cite</u> (URI), <u>datetime</u> (Datetime)		Flow (acepta elementos en línea y en bloque)	
Desarrollo de atributos			
cite <i>cite="original.html"</i>	El valor de este atributo es un URI que designa un documento o mensaje fuente, que sirve para hacer referencia a la información que explica por qué se modificó un documento. Distingue entre mayúsculas y minúsculas.		
datetime <i>datetime="2007-01-31T22:58:21DZH"</i>	El valor de este atributo especifica la fecha y la hora en que se hizo el cambio. Debe ser con el formato YYYY-MM-DDThh:mm:ssTZD donde la T indica la separación entre fecha, hora y zona horaria.		

Ejemplo

INS y DEL simples

CÓDIGO

```
<p>El locatario acepta abonar al locador la suma de <del>$500</del> <ins>$550</ins> el día <del>1</del> <ins>8</ins> de agosto de 2007.</p>
```

RESULTADO

El locatario acepta abonar al locador la suma de ~~\$500~~ \$550 el día ~~1~~ 8 de agosto de 2007.

🔗 [Ver ejemplo de INS y DEL \(CD > ejemplos > xhtml > texto > insdel.html\)](#)

INS y DEL incluyendo title, cite y datetime

CÓDIGO

```
<p>El locatario acepta abonar al locador la suma de <del>$500</del> <ins cite="detalleexpensas.html" datetime="2007-02-05T19:15-03:00">$550</ins> el día <del>1</del> <ins title="Fecha cambiada tras comunicación con el locatario">8</ins> de agosto de 2007.</p>
```

RESULTADO

El locatario acepta abonar al locador la suma de ~~\$500~~ \$550 el día ~~1~~ 8 de agosto de 2007.

Nota: Básicamente el resultado visual es exactamente igual al ejemplo anterior. Sin embargo, colocando el mouse sobre el número 8, sabemos que la fecha de pago se cambió tras una charla con el locatario.

Y accediendo al vínculo [detalleexpensas.html](#) veremos en concepto de qué servicios el alquiler fue aumentado \$50 el 5 de febrero del 2007.

🔗 [Ver ejemplo de INS y DEL con atributos \(CD > ejemplos > xhtml > texto > insdel1.html\)](#)

ELEMENTOS SUB Y SUP

Sintaxis: (etiqueta inicial y final obligatoria)

```
<sub> . . . </sub>
```

```
<sup> . . . </sup>
```

El elemento **SUB** se utiliza para insertar un **subíndice** en un texto y el elemento **SUP** para insertar un **superíndice**. Ambos permiten mostrar de forma correcta ecuaciones químicas, formulas matemáticas, notación científica y pies de página correctamente en una página web.

La mayoría de los agentes de usuario visuales reducen el tamaño del texto del superíndice y subíndice a la mitad, usando la misma fuente tipográfica que el resto del texto. Sin embargo la apariencia puede ser cambiada a través de hojas de estilo.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<p>Ejemplo de subíndice: 2H<sub>2</sub>+O<sub>2</sub> -> 2H<sub>2</sub>O</p>
<p>Ejemplo de superíndice: E = MC<sup>2</sup></p>
```

RESULTADO

Ejemplo de subíndice: 2H₂+O₂ -> 2H₂O

Ejemplo de superíndice: E = MC²

[Ver ejemplo de SUB y SUP \(CD > ejemplos > xhtml > texto > subsup.html\)](#)

ELEMENTO BDO (ANULACIÓN DEL ALGORITMO BIDIRECCIONAL)

Sintaxis: (etiqueta inicial y final obligatoria)

```
<bdo> . . . </bdo>
```

El elemento **BDO** es utilizado para cambiar la dirección de lectura del texto (anular el algoritmo bidireccional), de una lectura por defecto de izquierda a derecha, hacia otra de derecha a izquierda, o viceversa. Su propósito es permitir la correcta visualización de diferentes lenguajes en un mismo documento, es decir el español e inglés que se leen de izquierda a derecha, junto con el chino o hebreo que se leen de derecha a izquierda.

El algoritmo bidireccional y el atributo **dir** suelen ser suficientes para controlar los cambios de direccionalidad por inclusión, sin embargo, puede necesitarse el elemento **BDO** que permite a los autores desactivar el algoritmo

bidireccional para fragmentos de texto específicos, cuando se produce una presentación incorrecta.

Este elemento debería utilizarse cuando es preciso un control absoluto del orden de las secuencias (Ej. números de referencia multi-lenguaje), y el **atributo `dir` es siempre obligatorio**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir* (ltr rtl)		Inline (acepta elementos en línea)	
Desarrollo de atributos			
dir <code>dir="ltr"</code> <code>dir="rtl"</code>		Es un atributo obligatorio que especifica la dirección base del texto contenido en el elemento. Esta dirección prevalece sobre la direccionalidad inherente de los caracteres y los valores posibles son: <ul style="list-style-type: none"> • ltr: Texto de izquierda a derecha (Left-to-right). • rtl: Texto de derecha a izquierda (Right-to-left). 	

Ejemplo

CÓDIGO

```
<!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="en-US" lang="en-US">
<head>
  <title>Aplicación de BDO</title>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
  <meta name="author" content="Maira Vykus"/>
</head>

<body>
  <p>Este texto se lee con normalidad de izquierda a derecha. El siguiente texto
  debe verse invertido:</p>
  <p><bdo dir="rtl">Este texto debe invertirse.</bdo></p>
</body>
</html>
```

RESULTADO

El segundo párrafo debe invertirse:

Este texto se lee con normalidad de izquierda a derecha. El siguiente texto debe verse invertido:

.esnitrevni ebed otset etsE

[🔗 Ver ejemplo de BDO \(CD > ejemplos > xhtml > bdo.html\)](#)

SEPARACIÓN DE PALABRAS

Guión normal

El guión normal debería ser interpretado por un agente de usuario exactamente igual que cualquier otro carácter y se

representa mediante el carácter "-" (- o -).

Guión de separación

El guión de separación le dice al agente de usuario dónde puede producirse un salto de línea, y se representa mediante la referencia a entidad de caracteres ­ (­ o ­).

- Aquellos navegadores que interpreten los guiones de separación deben respetar la siguiente semántica:
- Si una línea se rompe por un guión de separación, debe mostrarse un guión al final de la primera línea.

Si no se produce un salto de línea en un guión de separación, el agente de usuario no debe mostrar el carácter de guión.

Es decir que, si a la palabra "interpretamos", le colocamos en el medio el guión de separación, resultaría en "interpre­tamos". Si la palabra queda ubicada en el medio de un renglón la misma será visualizada con normalidad, sin embargo, si queda al final de un renglón, al generarse el salto de línea, leeremos "interpre-" al final del renglón y "tamos" en el renglón siguiente.

ESPACIO EN BLANCO

En XHTML, sólo se definen como **caracteres de espacio en blanco** los siguientes:

- Espacio ()
- Tabulación ()
- Retorno de carro ()
- Avance de línea (
)
- Salto de página ()
- Espacio de anchura nula (​)

Los autores deberían utilizar los elementos y estilos apropiados, y no caracteres de espacio, para lograr efectos de formato visual relacionados con el espacio en blanco.

Para todos los elementos HTML, excepto para el elemento **PRE**, las secuencias de espacio en blanco separan "palabras" (con el término "palabra" queremos decir "secuencias de caracteres que no son de espacio en blanco"). Los agentes de usuario, al dar formato al texto, deberían identificar estas palabras y mostrarlas de acuerdo con las convenciones del idioma escrito y del medio destino.

Una secuencia de espacios en blanco entre palabras en el documento fuente puede ser representada como un espacio entre palabras completamente diferente (excepto en el caso del elemento **PRE**, usado para texto preformateado, en el cuál el espacio en blanco es significativo). En particular, los agentes de usuario deberían colapsar las secuencias de espacios en blanco de la entrada al producir la salida del espacio entre palabras.

Por otra parte los autores no deberían suponer que los agentes de usuario representarán el espacio en blanco que esté inmediatamente después de una etiqueta inicial o inmediatamente antes de una etiqueta final. Así, los autores, y en particular las herramientas de creación, deberían escribir:

```
<p>Ofrecemos <a>soporte técnico</a> gratuito a nuestros suscriptores.</p>
```

y no:

```
<p>Ofrecemos<a> soporte técnico </a>gratuito para nuestros suscriptores.</p>
```

6. Listas



Hay tres estructuras que permiten organizar la información en listas:

- Lista no ordenada
- Lista ordenada
- Lista de definiciones

Estos tres tipos de lista pueden anidarse entre sí. La apariencia de cada una varía según el agente de usuario, y se desaconseja el uso de listas para dar sangría al texto, ya que esta tarea puede conseguirse mediante el uso de hojas de estilo.

LISTA NO ORDENADA (UL), LISTA ORDENADA (OL) Y ELEMENTOS DE LISTA (LI)

Ambos tipos se representan de igual manera, donde cada ítem en la lista es encerrado en la etiqueta ``. La lista es separada del texto precedente y posterior mediante un quiebre de párrafo.

Estas listas pueden anidarse unas dentro de otras para lograr una jerarquía, donde cada diferente nivel tiene un icono y/o número que la identifica y un sangrado, según ese nivel de anidamiento.

Una **lista no ordenada** (unordered list), está delimitada en su inicio y final por el elemento **UL**. Este tipo de lista consiste en una serie de datos, relacionados de alguna manera y sin ningún orden en particular. Cada elemento de la lista es precedido por un icono.

Una **lista ordenada** (ordered list), es abierta y cerrada por el elemento **OL**. A diferencia de las listas no ordenadas, los datos deben estar listados en un orden en particular, y es por ello que los agentes de usuario visuales organizan los elementos numerándolos. Por ejemplo, este tipo de listas puede ser utilizado para un índice, tabla de contenido, instrucciones, etc. Por defecto la numeración comienza con el número 1, pero a través de CSS se puede especificar que la numeración sea con números romanos, letras, etc.

Elemento UL

Sintaxis: (etiqueta inicial y final obligatoria)

```
<ul>...</ul>
```

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout	compact ("compact"), type ("disc" "square" "circle")	(LI) + (una o más etiquetas LI)	

Elemento OL

Sintaxis: (etiqueta inicial y final obligatoria)

```
<ol>...</ol>
```

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout	compact ("compact"), start (Number), type ("arabic numbers" "lower alpha" "upper alpha" "lower roman" "upper roman")	(LI) + (una o más etiquetas LI)	

Elemento LI

Sintaxis: (etiqueta inicial y final obligatoria)

```
<li>...</li>
```

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout	type (CDATA), value (Number)	Flow (acepta elementos en línea o en bloque)	

La forma de representación visual de los elementos se puede especificar a través de hojas de estilo.

Ejemplos

Lista no ordenada

CÓDIGO

```
<ul>
  <li>Elemento 1</li>
  <li>Elemento 2</li>
</ul>
```

RESULTADO

- Elemento 1
- Elemento 2

[🔗 Ver ejemplo de lista desordenada \(CD > ejemplos > xhtml > listas > listas_desordenadas.html\)](#)

Lista ordenada

CÓDIGO

```
<ol>
  <li>Elemento 1</li>
  <li>Elemento 2</li>
</ol>
```

RESULTADO

1. Elemento 1
2. Elemento 2

[🔗 Ver ejemplo de lista ordenada \(CD > ejemplos > xhtml > listas > listas_ordenadas.html\)](#)

Lista anidada

CÓDIGO

```
<ol>
  <li>Elemento 1
    <ul>
      <li>Subelemento 1</li>
      <li>Subelemento 2</li>
      <li>Subelemento 3</li>
    </ul>
  </li>
  <li>Elemento 2</li>
</ol>
```

Nota: Note que la lista anidada es colocada enteramente dentro del primer elemento de la lista madre, es decir antes de la etiqueta de cierre `` de dicho elemento.

RESULTADO

1. Elemento 1
 - Subelemento 1
 - Subelemento 2
 - Subelemento 3
2. Elemento 2

[🔗 Ver ejemplo de listas anidadas \(CD > ejemplos > xhtml > listas > listas_anidadas.html\)](#)

LISTA DE DEFINICIÓN (DL, DT, DD)

Las **listas de definición** consisten en una pareja término/definición, donde un término o frase es seguido por una definición o explicación. En ellas, la etiqueta **dl** marca el inicio y el final de la lista, la etiqueta **dt** representa el término y permite sólo contenido en línea y por último, la definición está encerrada en una etiqueta **dd** que permite contenido en bloque.

La apariencia depende del agente de usuario (dispositivo o navegador), aunque generalmente el término es mostrado en el margen izquierdo y la definición con una sangría en el renglón siguiente. En algunos casos, si la definición es corta (3 o menos caracteres) la misma se coloca en la misma línea que el término. Toda la lista es separada del texto anterior y siguiente a través de un quiebre de párrafo.

Si bien las listas de definición suelen ser usadas para definir términos, otro uso sería para un diálogo, donde el término es el nombre del personaje y la definición lo que éste dice.

Elemento DL

Sintaxis: (etiqueta inicial y final obligatoria)

```
<dl> . . . </dl>
```

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout	compact ("compact")	(DT DD) + (una o más DT y DD en cualquier orden)	

Elemento DT

Sintaxis: (etiqueta inicial y final obligatoria)

```
<dt> . . . </dt>
```

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Elemento DD

Sintaxis: (etiqueta inicial y final obligatoria)

```
<dd> . . . </dd>
```

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Flow (acepta elementos en línea y en bloque)	

Ejemplo

CÓDIGO

```
<dl>
  <dt><em>Automóvil</em></dt>
  <dd>Que se mueve por sí mismo. Se dice principalmente de los vehículos que
  pueden ser guiados para marchar por una vía ordinaria sin necesidad de carriles y
  llevan un motor, generalmente de explosión, que los pone en movimiento.</dd>

  <dt><em>Casa</em></dt>
  <dd>Edificio para habitar</dd>
  <dd>Edificio de una o pocas plantas destinado a vivienda unifamiliar, en
  oposición a piso.</dd>
  <dd>Edificio, mobiliario, régimen de vida, etc., de alguien.</dd>

  <dt><em>Asiento</em></dt>
  <dt><em>Silla</em></dt>
  <dd>Mueble para sentarse.</dd>
</dl>
```

Nota: Tómese en cuenta que se ha encerrado con `` los términos para diferenciarlos de las definiciones y que a cada término puede corresponder una o más definiciones, así como una definición pueden tener uno o más términos asociados.

RESULTADO

Automóvil

Que se mueve por sí mismo. Se dice principalmente de los vehículos que pueden ser guiados para marchar por una vía ordinaria sin necesidad de carriles y llevan un motor, generalmente de explosión, que los pone en movimiento.

Casa

Edificio para habitar

Edificio de una o pocas plantas destinado a vivienda unifamiliar, en oposición a piso.

Edificio, mobiliario, régimen de vida, etc., de alguien.

Asiento

Silla

Mueble para sentarse.

[🔗 Ver ejemplo de listas de definición \(CD > ejemplos > xhtml > listas > listas_dedefinicion.html\)](#)

EJEMPLO DE LISTA ORDENADAS, NO ORDENADAS Y DE DEFINICIÓN ANIDADAS.

CÓDIGO


```
<dl>
  <dt><strong>Ingredientes:</strong></dt>
  <dd>
    <ul>
      <li>100 g de harina</li>
      <li>10 g de azúcar</li>
      <li>1 taza de agua</li>
      <li>2 huevos</li>
      <li>sal, pimienta</li>
    </ul>
  </dd>

  <dt><strong>Procedimiento:</strong></dt>
  <dd>
    <ol>
      <li>Mezcle los ingredientes secos.</li>
      <li>Agregue los ingredientes líquidos.</li>
      <li>Remueva durante 10 minutos.</li>
      <li>Hornear durante una hora a 300 grados.</li>
    </ol>
  </dd>

  <dt><strong>Nota:</strong></dt>
  <dd>Cocinar a fuego lento.<dd>
</dl>
```

RESULTADO

Ingredientes:

- 100 g de harina
- 10 g de azúcar
- 1 taza de agua
- 2 huevos
- sal, pimienta

Procedimiento:

1. Mezcle los ingredientes secos.
2. Agregue los ingredientes líquidos.
3. Remueva durante 10 minutos.
4. Hornear durante una hora a 300 grados.

Nota:

Cocinar a fuego lento.

[🔗](#) Ver ejemplo de listas anidadas (CD > ejemplos > xhtml > listas > listas_todasanidadadas.html)

7. Tablas



Una **tabla** es una presentación estructural de información usando filas y columnas.

Antes de comenzar a explicar la estructura básica de una tabla, con sus elementos y atributos, debemos aclarar, tal como lo indica la W3C en la Especificación de HTML 4.01 que: "no deberían usarse tablas con la única finalidad de organizar la presentación de los contenidos de un documento (es decir, de crear el "layout"), ya que esto puede ocasionar problemas cuando se represente en un medio no visual. Además, al incluir gráficos, estas tablas pueden forzar a los usuarios a hacer desplazar horizontalmente la pantalla para ver una tabla diseñada en un sistema con una pantalla más grande. Para minimizar estos problemas, los autores deberían usar hojas de estilo en lugar de tablas para organizar la presentación."⁽⁵⁾

Una tabla está formada por los siguientes elementos

- Tabla (table)
- Título (caption)
- Grupos de filas
- Encabezado (thead)
- Pie (tfoot)
- Cuerpo (tbody)
- Grupos de columnas (colgroup y col)
- Filas (tr)
- Celdas de encabezado (th)
- Celdas de datos (td)

 [Ver ejemplo de tabla completa sin estilos \(CD > ejemplos > xhtml > tabla > tabla_completa.html\)](#)

5 W3C - HTML 4.01 Specification - Tables (<http://www.w3.org/TR/html4/struct/tables.html>)

ELEMENTO TABLE

Sintaxis: (etiqueta inicial y final obligatoria)

```
<table>...</table>
```

El elemento **TABLE** es utilizado para designar que dicho elemento es una tabla, y una tabla es una presentación estructural de información usando filas y columnas. El interior de la tabla se construye utilizando los elementos **THEAD**, **TFOOT**, **TBODY**, **TR**, **TH**, **TD** y **CAPTION**.

Por defecto el flujo de texto e imágenes se detiene en el documento, la tabla se inserta en la siguiente línea y al finalizar la misma, el flujo de texto e imágenes se resumen en la línea posterior. Sin embargo, el uso de hojas de estilo puede permitir la colocación de una tabla entre los textos e imágenes.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout márgenes de celda: cellpadding (Length), cellspacing (Length) bordes y líneas de división: border (Pixels), frame ("void" "above" "below" "hsides" "lhs" "rhs" "vsides" "box" "border"), rules ("none" "groups" "rows" "cols" "all") otros: summary (Text), width (Length),	align ("left" "center" "right"), bgcolor (Color)	(CAPTION? , (COL* COLGROUP*), THEAD? , TFOOT? , (TBODY+ TR+))>	
Desarrollo de atributos			
summary summary="cadena de texto"	Indica a través de un texto, un resumen del propósito o estructura, especialmente para agentes de usuario que representen en medios no visuales como voz o Braille. Es una descripción más larga que la que brinda el elemento caption .		
width width="medida"	Indica el ancho de la tabla entera y está destinado a agentes de usuario visuales. Puede ser declarada como un número entero de pixels o como un porcentaje. Cuando el valor es porcentual, el mismo es relativo al espacio horizontal disponible del agente de usuario. En ausencia de una especificación, el ancho de la tabla lo determina el agente de usuario.		

En síntesis, una tabla acepta:

- cero o un elemento **CAPTION**, **THEAD**, **TFOOT**
- cero o más elementos **COL** y **COLGROUP**
- uno o más elementos **TBODY**

Representación incremental

El modelo de tablas permite a los agentes de usuario representar las mismas **incrementalmente**, es decir, a medida que llegan las filas de la tabla, en lugar de esperar a que lleguen todos los datos.

Para que el agente de usuario pueda dar formato a la tabla en un solo paso el autor debe brindar, a través de los elementos **COLGROUP** y **COL**:

- el número de columnas de la tabla
- el ancho de estas columnas

Si alguna de las columnas es especificada en términos relativos o porcentuales, entonces también se deberá especificar el ancho de la tabla.

Direccionalidad de las tablas

La direccionalidad de una tabla puede ser:

- la direccionalidad heredada (por defecto de izquierda a derecha)
- o la especificada por el atributo **dir** del elemento **TABLE**.

Para una tabla de izquierda a derecha, la columna cero es la del lado izquierdo y la fila cero es la superior. Para una columna de derecha a izquierda, la columna cero es la del lado derecho, y la fila cero es la superior.

Cuando un agente de usuario asigna celdas adicionales a una fila (Ver “Calculating the number of columns in a table” -<http://www.w3.org/TR/html4/struct/tables.html#h-11.2.4.3>), las celdas adicionales de la fila se añaden a la derecha de la tabla si la tabla es de izquierda a derecha, y a la izquierda si la tabla es de derecha a izquierda.

TABLE es el único elemento en el que **dir** invierte el orden visual de las columnas; no puede invertirse independientemente una fila individual (**tr**) ni un grupo de columnas (**colgroup**). Cuando el atributo **dir** se establece para un elemento **TABLE**, también afecta a la dirección del texto dentro de las celdas (ya que este atributo es heredado por elementos en bloque).

CÓDIGO

Para especificar una tabla de derecha a izquierda, el atributo **dir** se establece como sigue:

```
<table dir="rtl">
... el resto de la tabla ...
</table>
```

Nota: La dirección del texto en celdas individuales puede cambiarse estableciendo el atributo **dir** en un elemento que defina la celda.

ELEMENTO CAPTION

Sintaxis: (etiqueta inicial y final obligatoria)

```
<caption>...</caption>
```

El elemento **CAPTION**, a través de su texto describe la naturaleza de la tabla, es una descripción corta del propósito de la misma. Este elemento sólo se permite inmediatamente después de la etiqueta inicial de **TABLE** y cada tabla sólo puede tener un elemento **CAPTION**. Se pueden usar otras etiquetas y propiedades de estilo dentro de este elemento.

En los agentes de usuario visuales, este elemento no aparece dentro de la tabla, sino fuera, ya sea arriba (por defecto) o abajo y centrado con respecto a la tabla.

Los agentes de usuario visuales permiten a las personas con visión percibir rápidamente la estructura de una tabla a partir de los encabezados así como a partir del título. Sin embargo, el título no es suficiente como información para los agentes de usuario no visuales, por lo tanto, los autores deberían proporcionar información adicional resumiendo el propósito y la estructura de la tabla usando el atributo **summary** del elemento **TABLE**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout	align ("top" "bottom" "left" "right")	Inline (acepta elementos en línea)	

Ejemplo

CÓDIGO

```
<table border="1">
<caption>Etiquetas XHTML</caption>
<tr>
<td>Fila 1: Primer celda</td>
<td>Segunda celda</td>
</tr>
<tr>
<td>Fila 2: Primer celda</td>
<td>Segunda celda</td>
</tr>
</table>
```

RESULTADO

Etiquetas XHTML	
Fila 1: Primer celda	Segunda celda
Fila 2: Primer celda	Segunda celda

[Ver ejemplo de CAPTION \(CD > ejemplos > xhtml > tabla > caption.html\)](#)

ELEMENTOS THEAD, TFOOT Y TBODY

Sintaxis: (etiqueta inicial y final obligatoria)

```
<thead>...</thead>
<tfoot>...</tfoot>
<tbody>...</tbody>
```

Las filas de una tabla pueden ser agrupadas en:

- una cabecera de tabla (**THEAD**)
- un pie (**TFOOT**)
- y una o más secciones de cuerpo donde se muestra la información (**TBODY**)

Esto permite a los agentes de usuario desplazar la información de los cuerpos de la tabla independientemente de la cabecera y el pie. Además, cuando se imprimen tablas largas, la información de cabecera y pie puede repetirse en todas las páginas que contengan datos de la tabla.

Las secciones **THEAD**, **TFOOT** y **TBODY** debe contener al menos una fila (**TR**) y la misma cantidad de columnas. Las dos primeras deben tener información de las columnas, y el cuerpo por su parte, debe mostrar filas de datos.

El pie de tabla debe aparecer antes que el cuerpo, para que así los agentes de usuario puedan representarlo antes de recibir las potenciales filas de datos.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout alineación: valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)		(TR) + (una o más etiquetas TR)	

Ejemplo

CÓDIGO

```
<table border="1">
<caption>Etiquetas XHTML</caption>
  <thead>
    <tr>
      <th>Información de cabecera</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <td>Información de pie</td>
    </tr>
  </tfoot>
  <tbody>
    <tr>
      <td>Primera fila de datos del cuerpo</td>
    </tr>
    <tr>
      <td>Segunda fila de datos del cuerpo</td>
    </tr>
  </tbody>
</table>
```

RESULTADO

Etiquetas XHTML	
Información de cabecera	
Primera fila de datos del cuerpo	
Segunda fila de datos del cuerpo	
Información de pie	

Nota: En el encabezado se ha colocado una celda de encabezado (**TH**), en lugar de una celda de datos, para resaltar el significado.

🔗 [Ver ejemplo de grupo de filas \(CD > ejemplos > xhtml > tabla > grupodefila.html\)](#)

ELEMENTO COLGROUP

Sintaxis: (etiqueta inicial y final obligatoria)

```
<colgroup>...</colgroup>
```

El elemento **COLGROUP** asigna columnas a un grupo de columnas, permitiendo crear divisiones estructurales dentro de una tabla. Una tabla puede contener:

- un único grupo de columnas (cuando no se especifica un elemento **COLGROUP**).
- o cualquier número de grupos de columnas (cada uno delimitado por un elemento **COLGROUP**). Este número puede especificarse de dos maneras mutuamente exclusivas:
- El atributo **span** (valor por defecto 1) que especifica el número de columnas del grupo.
- El elemento **COL** que representa una o más columnas en el grupo y permite asignar valores a las columnas individuales dentro del elemento **COLGROUP**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout alineación: valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character) otros: span (Number), width (MultiLength)		(COL) * (CERO o más etiquetas COL)	
Desarrollo de atributos			
span span="númeroentero"	Este atributo, que debe ser un entero > 0, especifica el número de columnas de un grupo de columnas. Los diferentes valores significan lo siguiente: <ul style="list-style-type: none"> • En ausencia de un atributo span, cada COLGROUP define un grupo de columnas que contiene una columna. • Si el atributo span se establece en N > 0, el elemento COLGROUP actual define un grupo de columnas que contiene N columnas. Los agentes de usuario deben ignorar este atributo si el elemento COLGROUP contiene uno o más elementos COL . Además, si las columnas son desiguales, conviene usar COL para crear las mismas, en lugar de este atributo.		
width width="50px"	Este atributo especifica un ancho por defecto para cada columna del grupo de columnas actual. Además de los valores normales de píxeles, porcentajes y longitudes relativas, este atributo permite la forma especial "0*" (cero asterisco) que significa que el ancho de cada columna del grupo debería ser el ancho mínima necesaria para alojar los contenidos de la columna. Esto implica que antes de poder calcular correctamente el ancho de la columna deben conocerse primero todos los contenidos de la columna. Los autores deberían tener en cuenta que al especificar "0*" impedirán a los agentes de usuario visuales representar la tabla incrementalmente. El ancho especificado a una columna a través del elemento COL prevalece sobre el ancho especificado por el atributo width del elemento COLGROUP .		

El **ancho de las columnas** se puede especificar de tres maneras

- **Fijas (Fixed):** Una especificación de anchura fija está dada en píxeles (Ej. width="30"). Una especificación de anchura fija permite la representación incremental.

- **Porcentuales (Percentage):** Una especificación porcentual (Ej. `width="20%"`) se basa en el porcentaje del espacio horizontal disponible para la tabla (entre los bordes actuales izquierdo y derecho, incluyendo elementos flotantes). Obsérvese que este espacio no depende de la propia tabla, y por lo tanto las especificaciones porcentuales permiten la representación incremental.
- **Proporcionales (Proportional):** Las especificaciones proporcionales (Ej. `width="3*"`) se refieren a porciones de espacio horizontal requerido por una tabla. Si se da al ancho de la tabla un valor fijo por medio del atributo `width` del elemento `TABLE`, los agentes de usuario pueden representar la tabla incrementalmente incluso con columnas proporcionales. Sin embargo, si la tabla no tiene un ancho fijo, los agentes de usuario deben recibir todos los datos de la tabla antes de poder determinar el espacio horizontal requerido por la tabla. Sólo entonces puede asignarse este espacio a las columnas proporcionales.

Si no se especifica el ancho para una columna, el agente de usuario no puede dar formato a la tabla incrementalmente, ya que debe esperar a que lleguen todos los datos de la columna para poder asignarle un ancho adecuado. Si las anchuras de las columnas resultan ser demasiado estrechas para los contenidos de alguna de las celdas de la tabla, los agentes de usuario pueden optar por remodelar la tabla.

Nota: Aunque el atributo `width` del elemento `TABLE` no está desaprobado, se aconseja a los autores usar hojas de estilo para especificar anchuras de tabla.

Ejemplo

Aplicación de `colgroup`

CÓDIGO

Tal como indicamos, una tabla puede tener uno o más grupos de columnas. La siguiente tiene dos grupos de columnas (2 `COLGROUP`): el primer grupo formado por 6 columnas de 40px de ancho (de la celda 1 a la 6) y el segundo grupo formado por 3 columnas cuyo ancho será el mínimo necesario (de la celda 7 a la 9):

```
<table border="1">
<colgroup span="6" width="40px"></colgroup>
<colgroup span="3" width="0*"></colgroup>
<tr>
<td>Celda 1</td>
<td>Celda 2</td>
<td>Celda 3</td>
<td>Celda 4</td>
<td>Celda 5</td>
<td>Celda 6</td>
<td>Celda 7</td>
<td>Celda 8</td>
<td>Celda 9</td>
</tr>
<tr>
<td>Celda 1</td>
<td>Celda 2</td>
<td>Celda 3</td>
<td>Celda 4</td>
<td>Celda 5</td>
<td>Celda 6</td>
<td>Celda 7</td>
<td>Celda 8</td>
<td>Celda 9</td>
</tr>
</table>
```

RESULTADO

Celda 1	Celda 2	Celda 3	Celda 4	Celda 5	Celda 6	Celda 7	Celda 8	Celda 9
Celda 1	Celda 2	Celda 3	Celda 4	Celda 5	Celda 6	Celda 7	Celda 8	Celda 9

[Ver ejemplo de grupo de columnas \(CD > ejemplos > xhtml > tabla > colgroup.html\)](#)

Utilización de span en lugar de COL

Utilizando el atributo **span** se puede agrupar información sobre el ancho de las columnas, por lo cuál, si una tabla tiene 5 columnas de 10 píxeles de ancho se podría escribir:

```
<colgroup span="5" width="10px"></colgroup>
```

En lugar de:

```
<colgroup>
  <col width="10px" />
  <col width="10px" />
  <col width="10px" />
  <col width="10px" />
  <col width="10px" />
</colgroup>
```

Nota: En este caso, con 5 columnas únicamente no se aprecia la magnitud del problema, sin embargo, si fueran 40 columnas, sería innecesariamente largo escribir tantas etiquetas **COL**.

ELEMENTO COL

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<col />
```

El elemento **COL** permite asignar atributos a las columnas dentro de un grupo de columnas, sin que ello implique ningún tipo de agrupamiento estructural. Los elementos **COL** están vacíos y sólo sirven como soporte de atributos, ya que su única función es indicar el número de columnas que compartirán los atributos del elemento.

El elemento **COL** sólo puede aparecer dentro de un elemento **COLGROUP**. Sin embargo, la etiqueta **COL** debe ser omitida si se usa el atributo **span** del elemento **COLGROUP**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout alineación: valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character) otros: <u>span</u> (Number), <u>width</u> (MultiLenght)		Empty	
Desarrollo de atributos			

span span="númeroentero"	Este atributo, cuyo valor debe ser un entero > 0, especifica el número de columnas que "abarca" el elemento COL ; quien comparte sus atributos con todas las columnas que abarca. El valor por defecto de este atributo es 1 (es decir, el elemento COL se refiere a una sola columna). Si el atributo span se establece en $N > 1$, el elemento COL actual comparte sus atributos con las siguientes $N-1$ columnas.
width width="50px"	Este atributo especifica un ancho por defecto para todas las columnas abarcadas por el elemento COL actual. Tiene el mismo significado que el atributo width del elemento COLGROUP y prevalece sobre él. Este valor puede ser en pixels o porcentajes.

Ejemplo

CÓDIGO

Cuando es necesario distinguir una o más columnas del resto (con un ancho o estilo específico), se puede identificar esa columna con un elemento **COL**. Por consiguiente, para aplicar un estilo especial a la última columna del ejemplo anterior, la singularizamos del siguiente modo:

```
<table border="1">
<colgroup width="50px">
  <col span="4" />
  <col width="150px" />
</colgroup>
<tr>
  <td>Celda 1</td>
  <td>Celda 2</td>
  <td>Celda 3</td>
  <td>Celda 4</td>
  <td>Celda 5</td>
</tr>
<tr>
  <td>Celda 1</td>
  <td>Celda 2</td>
  <td>Celda 3</td>
  <td>Celda 4</td>
  <td>Celda 5</td>
</tr>
</table>
```

El atributo **width** del elemento **COLGROUP** es heredado por las 5 columnas. El primer elemento **COL** se refiere a las primeras 4 columnas (sin darle ningún atributo) y el segundo **COL** asigna a la quinta columna un **width** de 150px (que se impone al **width** asignado por el **COLGROUP**).

RESULTADO

Celda 1	Celda 2	Celda 3	Celda 4	Celda 5
Celda 1	Celda 2	Celda 3	Celda 4	Celda 5

[Ver ejemplo de columnas \(CD > ejemplos > xhtml > tabla > col.html\)](#)

ELEMENTO TR

Sintaxis: (etiqueta inicial y final obligatoria)

```
<tr>...</tr>
```

La etiqueta **TR** es usada para crear una fila dentro de un elemento **TABLE** y se pueden crear tantas filas como se deseen.

Una fila puede contener una o más celdas creadas con **TH** o **TD**, las cuáles pueden contener cualquier tipo de información o imágenes. Una etiqueta **TR** nunca debe aparecer dentro de otra **TR** o dentro de **TD** o **TH**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout alineación: valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)	bicolor (Color)	(TH TD) + (uno o más elementos TH y TD)	

Ejemplo

CÓDIGO

Una tabla podría tener tres filas de la siguiente manera:

```
<table border="1">
  <tr><th>Fila de encabezado</th></tr>
  <tr><td>Primer fila de contenido</td></tr>
  <tr><td>Segunda fila de contenido</td></tr>
</table>
```

RESULTADO

Fila de encabezado
Primer fila de contenido
Segunda fila de contenido

Nota: Cada declaración de **TR** establece una nueva fila, por lo que tres declaraciones, son tres filas.

🔗 [Ver ejemplo de TR \(CD > ejemplos > xhtml > tabla > tr.html\)](#)

ELEMENTOS TH Y TD

Sintaxis: (etiqueta inicial y final obligatoria)

```
<th>...</th> (celda de encabezado)
<td>...</td> (celda de datos)
```

Una tabla puede contener **celdas de encabezado** y **celdas de datos**. Esta distinción permite a los agentes de usuario representar la información de manera diferente, incluso en ausencia de hojas de estilo. Por ejemplo, las celdas de encabezado pueden ser presentadas por los agentes de usuario visuales con una fuente en negrita, y por los sintetizadores de voz con una inflexión de voz diferente.

El elemento **TH** es utilizado para crear una celda de encabezado en una fila. Estas son mostradas en un estilo de fuente en negrita y proveen de un título, nombre o información a las columnas de celdas. Los agentes de usuario disponen

de los contenidos del elemento **TH** y el valor del atributo **abbr**, y deben presentar al usuario la información de alguno de ellos. En medios visuales, el último puede ser apropiado cuando no hay espacio suficiente para representar los contenidos completos de la celda. Para medios no visuales puede usarse **abbr** como abreviatura de los encabezados de la tabla cuando éstos se representen junto con los contenidos de las celdas a las que se aplican.

Las etiquetas **TD** son usadas para crear celdas que contienen la información que se desea mostrar en la tabla. Se pueden colocar tantas celdas como se deseen en una fila, y las mismas pueden estar vacías (es decir, pueden no contener datos).

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout alineación: valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character) otros: abbr (Text), axis (CDATA), headers (IDREFS), scope ("row" "col" "rowgroup" "colgroup"), rowspan (Number), colspan (Number)	bgcolor (Color), height (Length), nowrap ("nowrap"), width (Length)	Flow (acepta elementos en línea y en bloque)	
Desarrollo de atributos			
headers <i>headers="nombre1,nombre2"</i>	Lista de id's de celdas de encabezado. No es reconocido por la mayoría de los navegadores. Especifica la lista de celdas de encabezado que proporcionan información de encabezado para la celda de datos actual. El valor de este atributo es una lista de nombres de celdas separados por comas. Los nombres surgen del atributo id de las celdas. Generalmente los autores usan el atributo headers para ayudar a los agentes de usuario no visuales a representar información de encabezado sobre celdas de datos (Ej. leyendo la información de encabezado antes de los datos de la celda), pero el atributo también puede emplearse junto con hojas de estilo.		
scope <i>scope="row"</i> <i>scope="col"</i> <i>scope="rowgroup"</i> <i>scope="colgroup"</i>	Es utilizado para asignar un conjunto de celdas de información a una celda de encabezado. No es reconocido por la mayoría de los navegadores. Para las tablas normales scope es más simple que el atributo headers . Este atributo puede usarse en lugar del atributo headers , en particular en tablas sencillas. Si se especifica, este atributo debe tener uno de los siguientes valores: <ul style="list-style-type: none"> • row (fila): La celda actual proporciona información de encabezado para el resto de la fila que la contiene. • col (columna): La celda actual proporciona información de encabezado para el resto de la columna que la contiene. • rowgroup (grupo de filas): La celda de encabezado proporciona información de encabezado para el resto del grupo de filas que la contiene. • colgroup (grupo de columnas): La celda de encabezado proporciona información de encabezado para el resto del grupo de columnas que la contiene. 		

abbr <i>abbr="Art."</i>	No es reconocido por la mayoría de los navegadores. Este atributo debería usarse para proporcionar una forma abreviada del contenido de la celda; los agentes de usuario pueden representar esta forma abreviada en lugar del contenido de la celda cuando sea apropiado. Los nombres abreviados deberían ser cortos, ya que los agentes de usuario pueden representarlos repetidas veces. Por ejemplo, los sintetizadores de voz pueden representar los encabezados abreviados relacionados con una celda en particular antes de representar el contenido de esa celda.
axis <i>axis="ciudades"</i>	No es reconocido por la mayoría de los navegadores. Este atributo se utiliza para situar una celda en categorías conceptuales, las cuales pueden considerarse como ejes de un espacio n-dimensional. Los agentes de usuario pueden dar a los usuarios acceso a estas categorías (Ej. el usuario puede pedir al agente de usuario todas las celdas que pertenecen a ciertas categorías, el agente usuario puede presentar una tabla con la forma de una tabla de contenidos, etc.). El valor de este atributo es una lista de nombres de categorías separados por comas.
rowspan <i>rowspan="n"</i>	Este atributo especifica el número de filas (dos o más) abarcado por la celda actual. El valor por defecto de este atributo es uno ("1"). El valor cero ("0") significa que la celda abarca todas las filas desde la fila actual hasta la última fila de la sección (thead , tbody , o tfoot) en la que la celda está definida.
colspan <i>colspan="n"</i>	Este atributo especifica el número de columnas (dos o más) abarcado por la celda actual. El valor por defecto de este atributo es uno ("1"). El valor cero ("0") significa que la celda abarca todas las columnas desde la columna actual hasta la última columna del grupo de columnas (colgroup) en que la celda está definida.

Ejemplos

Celdas de encabezado y datos

CÓDIGO

La siguiente tabla contiene 3 columnas de datos, cada una encabezada por una descripción de la columna.

```
<table border="1" summary="Esta tabla muestra el número y tipo de bolitas ganadas
por los niños del barrio">
<caption>Bolitas ganadas</caption>
<tr>
<th>Nombre</th>
<th>Cantidad de bolitas</th>
<th>Tipo de bolitas</th>
</tr>
<tr>
<td>Sebastian</td>
<td>10</td>
<td>Pequeñas</td>
</tr>
<tr>
<td>Mariano</td>
<td>2</td>
<td>Grandes</td>
</tr>
</table>
```

RESULTADO

En un navegador estándar:

Bolas ganadas		
Nombre	Cantidad de bolitas	Tipo de bolitas
Sebastian	10	Pequeñas
Mariano	2	Grandes

En un dispositivo tty:

Nombre	Cantidad de bolitas	Tipo de bolitas
Sebastian	10	Pequeñas
Mariano	2	Grandes

[Ver ejemplo de TH y TD 1 \(CD > ejemplos > xhtml > tabla > thtd1.html\)](#)

Celdas que abarcan varias columnas (COLSPAN)

CÓDIGO

Las celdas pueden abarcar varias columnas, y la cantidad abarcada es definida por el atributo **colspan** de los elementos **th** y **td**.

En la siguiente tabla, se especifica que la primera celda de la fila 2, debe abarcar dos columnas. En base a esto, el segundo **TD** (F) hace referencia a la tercera columna.

```
<table border="1">
  <tr>
    <td>A</td>
    <td>B</td>
    <td>C</td>
  </tr>
  <tr>
    <td colspan="2">D</td>
    <td>F</td>
  </tr>
  <tr>
    <td>G</td>
    <td>H</td>
    <td>I</td>
  </tr>
</table>
```

RESULTADO

En un navegador estándar:

A	B	C
D	F	
G	H	I

En un dispositivo tty:

A	B	C
D		F
G	H	I

🔗 [Ver ejemplo de TH y TD 2 \(CD > ejemplos > xhtml > tabla > thtd2.html\)](#)

Celdas que abarcan varias filas (ROWSPAN)

CÓDIGO

Las celdas pueden abarcar varias filas, y la cantidad abarcada es definida por el atributo **rowspan** de los elementos **th** y **td**.

Aquí le indicamos a la tabla que la segunda celda de la primera fila debe abarcar dos filas:

```
<table border="1">
  <tr>
    <td>A</td>
    <td rowspan="2">B</td>
    <td>C</td>
  </tr>
  <tr>
    <td>D</td>
    <td>F</td>
  </tr>
  <tr>
    <td>G</td>
    <td>H</td>
    <td>I</td>
  </tr>
</table>
```

Al abarcar la celda "B" la primera y la segunda fila, la definición de la segunda fila sólo definirá dos celdas, la primera y la tercera.

RESULTADO

En un navegador estándar:

A	B	C
D		F
G	H	I

En un dispositivo tty:

A	B	C
D		F
G	H	I

Si el **TD** que define la celda "F" hubiera sido omitido, el agente de usuario habría añadido una celda adicional vacía para completar la fila.

[Ver ejemplo de TH y TD 3 \(CD > ejemplos > xhtml > tabla > thtd3.html\)](#)

Celdas superpuestas

CÓDIGO

La definición de celdas superpuestas es un error y los agentes de usuario pueden tratar este error de formas diferentes (Ej. la representación puede variar).

La superposición se genera cuando al asignar el atributo **rowspan** y **colspan** a diferentes celdas, las fusiones de dichas celdas sean coincidentes.

Por ejemplo, en la siguiente tabla la celda "E" abarca dos filas y la celda "G" abarca dos columnas, de modo que hay una superposición en la celda que está entre "G" y "H":

```
<table border="1">
  <tr>
    <td>A</td>
    <td>B</td>
    <td>C</td>
  </tr>
  <tr>
    <td>D</td>
    <td rowspan="2">E</td>
    <td>F</td>
  </tr>
  <tr>
    <td colspan="2">G</td>
    <td>H</td>
  </tr>
</table>
```

RESULTADO

Este es un posible resultado en Explorer:

A	B	C
D	E	F
G		H

[Ver ejemplo de TH y TD 4 \(CD > ejemplos > xhtml > tabla > thtd4.html\)](#)

ATRIBUTOS DE TABLAS

Actualmente además de los atributos abajo explicados, se puede utilizar las propiedades de CSS2 para dar formato visual a las tablas.

Los atributos que permiten en HTML dar formato a las tablas son:

- los estilos de los bordes (**border**, **frame** y **rules**)
- el alineación horizontal y vertical de los contenidos de las celdas (**align**, **valign**, **char** y **charoff**)
- y los márgenes de las celdas (**cellspacing** y **cellpadding**)

Estilos de borde

frame

Sintaxis: `frame="void"`

Especifica los **bordes externos**, cuáles de los cuatro lados del marco que rodea a una tabla serán visibles. Los posibles valores son:

- **void**: Ningún lado. Este es el valor por defecto.
- **above**: Sólo el borde superior.
- **below**: Sólo el borde inferior.
- **hsides**: Sólo los bordes superior e inferior.
- **vsides**: Sólo los lados derecho e izquierdo.
- **lhs**: Sólo el lado izquierdo.
- **rhs**: Sólo el lado derecho.
- **box**: Los cuatro lados.
- **border**: Los cuatro lados.

rules

Sintaxis: `rules="groups"`

Este atributo especifica qué **líneas de división interiores** aparecerán entre las celdas de una tabla. La representación de las líneas de división depende del agente de usuario. Valores posibles:

- **none**: Ninguna línea de división. Este es el valor por defecto.
- **groups**: Sólo aparecerán líneas de división entre grupos de filas (**THEAD**, **TFOOT** y **TBODY**) y grupos de columnas (**COLGROUP** y **COL**).
- **rows**: Sólo aparecerán líneas de división entre filas.
- **cols**: Sólo aparecerán líneas de división entre columnas.
- **all**: Aparecerán líneas de división entre todas las filas y columnas.

border

Sintaxis: `border="1px"`

Este atributo especifica el ancho (sólo en píxeles) del marco que rodea a una tabla. Por defecto no hay un border (0 pixels).

Ejemplos

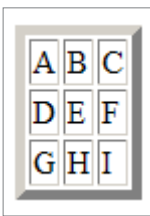
Border

CÓDIGO

Para ayudar a distinguir las celdas de una tabla, se puede establecer el atributo **border** del elemento **TABLE**. En el ejemplo se deberían mostrar bordes de 5 píxeles.

```
<table border="5">
  <tr>
    <td>A</td>
    <td>B</td>
    <td>C</td>
  </tr>
  <tr>
    <td>D</td>
    <td>E</td>
    <td>F</td>
  </tr>
  <tr>
    <td>G</td>
    <td>H</td>
    <td>I</td>
  </tr>
</table>
```

RESULTADO



A	B	C
D	E	F
G	H	I

[Ver ejemplo de border en tablas \(CD > ejemplos > xhtml > tabla > atributos_border.html\)](#)

Border, frame y rules

CÓDIGO

Si al elemento **TABLE** del ejemplo anterior le agregamos los atributos **frame="vsides"** **rules="cols"**, los 5 píxeles de borde serían en el lado derecho izquierdo y derecho de la tabla (**frame="vsides"**) con líneas de división entre cada columna.

```
<table border="5" frame="vsides" rules="cols">
  <tr>
    <td>A</td>
    <td>B</td>
    <td>C</td>
  </tr>
  <tr>
    <td>D</td>
    <td>E</td>
    <td>F</td>
  </tr>
  <tr>
    <td>G</td>
    <td>H</td>
    <td>I</td>
  </tr>
</table>
```

RESULTADO

A	B	C
D	E	F
G	H	I

🔗 [Ver ejemplo de border, frames y rules \(CD > ejemplos > xhtml > tabla > atributos_border_frames_rules.html\)](#)

Equivalencias

Los agentes de usuario y autores deben tener en cuenta que:

- `border="0"` es equivalente a `frame="void"` y `rules="none"`.
- `border > 0` implica `frame="border"` y, a menos que se especifique lo contrario, `rules="all"`.

En base a esto, las siguientes definiciones son equivalentes:

```
<table border="2">
<table border="2" frame="border" rules="all">
```

Nota: El atributo `border` también define el comportamiento de los bordes de los elementos `OBJECT` e `IMG`, pero toma valores diferentes para esos elementos.

Alineación horizontal y vertical

`align`

Sintaxis: `align="center"`

Especifica la **alineación horizontal** de los datos y la **justificación del texto** de una celda. Valores posibles:

- `left`: Datos a la izquierda/Texto justificado a la izquierda.
- `center`: Datos centrados/Texto con justificación centrada. Este es el valor por defecto para los encabezados de las tablas.
- `right`: Datos a la derecha/Texto justificado a la derecha.
- `justify`: Texto doblemente justificado. Los agentes de usuario que no soporten este valor deberían usar el valor heredado de la direccionalidad.
- `char`: Alinear el texto alrededor de un carácter específico. Si un agente de usuario no soporta alineación alrededor de un carácter, el comportamiento en presencia de este valor queda sin especificar.

`valign`

Sintaxis: `valign="top"`

Especifica la **alineación vertical de los datos** dentro de una celda. Valores posibles:

- `top`: Los datos de la celda se alinean con la parte superior de la celda.
- `middle`: Los datos de la celda se centran verticalmente dentro de la celda. Este es el valor por defecto.
- `bottom`: Los datos de la celda se alinean con la parte inferior de la celda.
- `baseline`: Todas las celdas que estén en la misma fila que una celda cuyo atributo `valign` tenga este valor deberían tener sus datos textuales posicionados de tal modo que la primera línea de texto aparezca en una línea de base común para todas las celdas de la fila. Esta restricción no se aplica a las líneas subsiguientes de texto de estas celdas.

`char`

Sintaxis: `char="."`

Este atributo especifica un carácter individual que actúa como eje de alineación. El valor por defecto es el carácter de

punto decimal para el idioma en uso, definido por el atributo **lang** (Ej. el punto (".") en inglés y la coma (",") en francés). Los agentes de usuario no necesitan soportar este atributo. Esto permite alinear columnas con precios, en base a su punto decimal.

charoff

Sintaxis: *charoff*="20px"

Si está presente, este atributo especifica la distancia (offset) entre el borde y la primera aparición del carácter de alineación en cada línea. Si una línea no incluye el carácter de alineación, debería ser desplazada horizontalmente hasta la posición de alineación.

Cuando se usa **charoff** para establecer el offset de un carácter de alineación, la dirección del desplazamiento está determinada por la dirección actual del texto (establecida con el atributo **dir**). En texto de izquierda a derecha (el valor por defecto), el desplazamiento es desde el margen izquierdo. En textos de derecha a izquierda, el desplazamiento es desde el margen derecho. Los agentes de usuario no necesitan soportar este atributo.

Herencia de las especificaciones de alineación

La alineación de los contenidos puede ser especificada para cada celda, sin embargo, la misma también puede ser heredada de la fila, columna o tabla.

Por lo tanto, en la alineación el orden de preferencia de mayor a menor depende de un atributo declarado en diferentes elementos, y ese orden varía según la alineación sea horizontal o vertical:

	Alineación Horizontal (align , char y charoff)	Alineación Vertical (valign)
1	Un elemento dentro de la celda (Ej. P)	
2	Una celda (TH y TD)	
3	Un elemento de grupo de columnas (COL y COLGROUP). Cuando la celda sea parte de un tramo que abarque varias columnas, la propiedad de alineación se hereda de la definición de la celda al comienzo del tramo	Un elemento de fila o grupo de filas (TR , THEAD , TFOOT y TBODY). Cuando una celda sea parte de un tramo que abarque varias filas, la propiedad de alineación se hereda de la definición de la celda al comienzo del tramo
4	Un elemento de fila o grupo de filas (TR , THEAD , TFOOT y TBODY). Cuando una celda sea parte de un tramo que abarque varias filas, la propiedad de alineación se hereda de la definición de la celda al comienzo del tramo	Un elemento de grupo de columnas (COL y COLGROUP). Cuando la celda sea parte de un tramo que abarque varias columnas, la propiedad de alineación se hereda de la definición de la celda al comienzo del tramo
5	la tabla (TABLE)	
6	el valor por defecto de la alineación	

Además, en las celdas:

- al determinar la alineación horizontal las columnas tienen preferencia sobre las filas, y
- al determinar la alineación vertical las filas tienen preferencia sobre las columnas.

Siempre debe recordarse que una celda puede heredar un atributo no de su padre sino de la primera celda de un tramo.

Ejemplo

CÓDIGO

La siguiente tabla alinea valores monetarios de una columna en base al punto decimal.

```

<table border="1">
<colgroup>
  <col />
  <col align="char" char="." />
</colgroup>
<thead>
  <tr>
    <th>Clientes</th>
    <th>Importe</th>
  </tr>
</thead>
<tbody>
  <tr>
    <td>Mariano Gomez</td>
    <td>$5</td>
  </tr>
  <tr>
    <td>Sebastian Messi</td>
    <td>$40.80</td>
  </tr>
  <tr>
    <td>Alejandra Pateri</td>
    <td>$170.20</td>
  </tr>
</tbody>
</table>

```

RESULTADO

Con lo cuál el resultado sería algo similar a la siguiente tabla (tener en cuenta que no todos los navegadores alinearán los importes, ej. Explorer y Firefox no lo hacen.):

Cliente	Importe
Mariano Gomez	\$1
Sebastián Messi	\$10.50
Alejandra Pateri	\$100.30

🔗 [Ver ejemplo de char y charoff \(CD > ejemplos > xhtml > tabla > atributos_border_char_charoff.html\)](#)

Cuando en el texto de una celda el carácter de alineación aparece más de una vez y los contenidos no caben en una sola línea, el comportamiento del agente de usuario queda sin especificar. Hay que ser precavido al utilizar este atributo.

Márgenes de las celdas

cellspacing

Sintaxis: *cellspacing="2px"*

Este atributo especifica cuánto espacio debería ser dejado por el agente de usuario entre celdas y entre las celdas individuales y el borde exterior (superior, inferior, derecho e izquierdo). Este valor se declara en pixels, y por defecto son 2 pixels de distancia.

cellpadding

Sintaxis: *cellpadding="5px"*

Este atributo especifica la cantidad de espacio entre el borde de la celda y sus contenidos. Si el valor de este atributo es una longitud en píxeles, los cuatro bordes deberían estar a esta distancia de los contenidos. Si el valor del atributo

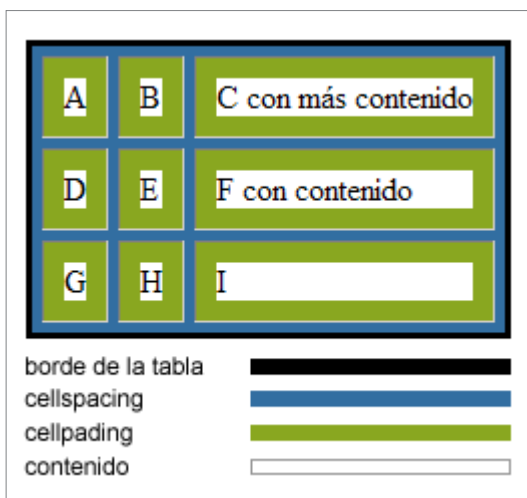
es una longitud porcentual, los bordes superior e inferior deberían estar igualmente separados del contenido según un porcentaje del espacio vertical disponible, y los bordes izquierdo y derecho deberían estar igualmente separados de los contenidos según un porcentaje del espacio horizontal disponible. Por defecto la distancia es 1 píxel.

Ejemplo

CÓDIGO

```
<table border="1" cellspacing="5px" cellpadding="10%">
  <tr>
    <td>A</td>
    <td>B</td>
    <td>C</td>
  </tr>
  <tr>
    <td>D</td>
    <td>E</td>
    <td>F</td>
  </tr>
  <tr>
    <td>G</td>
    <td>H</td>
    <td>I</td>
  </tr>
</table>
```

RESULTADO



En base al código existente:

- `cellspacing="5px"` indica que las celdas debe estar separadas entre sí y hasta el marco de la tabla por veinte píxeles.
- `cellpadding="20%"` especifica que el margen superior e inferior de la celdas estarán separados de sus contenidos por el 10% del espacio vertical disponible (para un total del 20%). Así mismo, el borde izquierdo y el borde derecho estarán separados de los contenidos por el 10% del espacio horizontal disponible (para un total del 20%).

Nota: En el archivo HTML se agregaron párrafos dentro de las celdas para poder identificar cada sector con un estilo diferente.

[Ver ejemplo de cellspacing y cellpadding \(CD > ejemplos > xhtml > tabla > atributos_cellpadding_cellspacing.html\)](#)

Nota: Ver Anexo > [Tablas y agentes de usuario no visuales](#) para más información sobre la representación de las tablas

por los agentes de usuario no visuales.

EJEMPLO DE TABLA COMPLETA

La siguiente tabla, es una comparativa de planes de hosting:

CÓDIGO

```
<table frame="box" rules="groups" border="1"
summary="Tabla comparativa de planes de hosting simple, con características y
precios.">
<caption>Planes de hosting simple</caption>
<colgroup align="left"></colgroup>
<colgroup align="center" span="4"></colgroup>
  <thead>
    <tr>
      <th>TARIFAS</th>
      <th>50MB</th>
      <th>100MB</th>
      <th>200MB</th>
      <th>300MB</th>
    </tr>
    <tr>
      <td>Costo de instalación</td>
      <td>Gratis</td>
      <td>Gratis</td>
      <td>Gratis</td>
      <td>Gratis</td>
    </tr>
    <tr>
      <td>Costo mensual</td>
      <td>$5</td>
      <td>$10</td>
      <td>$15</td>
      <td>$20</td>
    </tr>
    <tr>
      <td>Periodicidad de pago mínima</td>
      <td>Anual</td>
      <td>Semestral</td>
      <td>Mensual</td>
      <td>Mensual</td>
    </tr>
    <tr>
      <td>Costo total (pago mínimo)</td>
      <td>$60</td>
      <td>$60</td>
      <td>$15</td>
      <td>$20</td>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>CARACTERÍSTICAS</th>
      <th colspan="4"></th>
    </tr>
    <tr>
      <td>Espacio en disco</td>
      <td>50MB</td>
      <td>100MB</td>
      <td>200MB</td>
      <td>300MB</td>
    </tr>
```

```

</tr>
<tr>
  <td>Transferencia</td>
  <td>2GB</td>
  <td>5GB</td>
  <td>10GB</td>
  <td>15GB</td>
</tr>
<tr>
  <td>Dominio propio</td>
  <td colspan="4">Si</td>
</tr>
<tr>
  <td>Dominios alias</td>
  <td>2</td>
  <td>3</td>
  <td>4</td>
  <td>5</td>
</tr>
<tr>
  <td>Cuentas de correo POP3</td>
  <td>25</td>
  <td>50</td>
  <td>100</td>
  <td>150</td>
</tr>
<tr>
  <td>Aliases de correo</td>
  <td colspan="4">Ilimitados</td>
</tr>
</tbody>
<tbody>
  <tr>
    <th>TECNOLOGÍAS</th>
    <th colspan="4"></th>
  </tr>
  <tr>
    <td>Cuentas FTP</td>
    <td>25</td>
    <td>50</td>
    <td>100</td>
    <td>150</td>
  </tr>
  <tr>
    <td>Administrador MySQL - phpMyAdmin</td>
    <td colspan="4">Si</td>
  </tr>
  <tr>
    <td>Panel de Control Web</td>
    <td colspan="4">Si</td>
  </tr>
  <tr>
    <td>PHP</td>
    <td colspan="4">Si</td>
  </tr>
  <tr>
    <td>Base de datos MySQL</td>
    <td colspan="4">Si</td>
  </tr>
  <tr>
    <td>Extensiones Frontpage 2002</td>
    <td colspan="4">Si</td>
  </tr>
</tbody>
</table>

```


RESULTADO

[Ver ejemplo de tabla completa \(CD > ejemplos > xhtml > tabla > tabla_completa.html\)](#)

Planes de hosting simple

TARIFAS	50MB	100MB	200MB	300MB
Costo de instalación	Gratis	Gratis	Gratis	Gratis
Costo mensual	\$5	\$10	\$15	\$20
Periodicidad de pago mínima	Anual	Semestral	Mensual	Mensual
Costo total (pago mínimo)	\$60	\$60	\$15	\$20
CARACTERÍSTICAS	50MB	100MB	200MB	300MB
Espacio en disco	2GB	5GB	10GB	15GB
Transferencia				
Dominio propio			Si	
Dominios alias	2	3	4	5
Cuentas de correo POP3	25	50	100	150
Alias de correo			Ilimitados	
TECNOLOGÍAS	50MB	100MB	200MB	300MB
Cuentas FTP	25	50	100	150
Administrador MySQL - phpMyAdmin			Si	
Panel de Control Web			Si	
PHP			Si	
Base de datos MySQL			Si	
Extensiones Frontpage 2002			Si	

8. Vínculos



Un **vínculo**, **enlace** o **hipervínculo** es una conexión desde un recurso web a otro y ha sido una de las principales fuerzas que ha hecho posible el éxito de la Web.

Un **vínculo** tiene dos extremos o anclas (anchors) y una dirección. El vínculo comienza en el "ancla de origen" (origen del vínculo) y apunta al "ancla destino" (destino del vínculo), que puede ser cualquier recurso de la Web (Ej. una imagen, un videoclip, un archivo de sonido, un programa, un documento HTML, un elemento dentro de un documento HTML, etc.).

Esos recursos se obtienen seleccionando un vínculo con un clic del mouse o a través del teclado. En el siguiente ejemplo, "Clarín" es el origen del vínculo y el atributo **href** especifica la dirección del destino del vínculo.

```
<a href="http://www.clarin.com.ar">Clarín</a>
```

La referencia a un recurso no disponible o no identificable es un error.

Si bien algunos elementos y atributos HTML (**IMG**, **FORM**) crean vínculos a otros recursos, en este capítulo sólo haremos referencia a los vínculos creados con los elementos **A** y **LINK**.

IDENTIFICADOR UNIVERSAL DE RECURSOS (URI)

Todos los recursos disponibles en la Web (documentos HTML, imágenes, videos, programas, etc.) tienen una dirección que puede ser codificada mediante un URI (Universal Resource Identifier), o **Identificador Universal de Recursos**.

Los URIs son utilizados para:

- Crear un vínculo a otro documento o recurso (**A** y **LINK**).
- Crear un vínculo a una hoja de estilo o script externo (**LINK** y **SCRIPT**)
- Incluir una imagen, objeto o aplicación (**IMG**, **OBJECT** e **INPUT**).

- Crear un mapa de imágenes (**MAP** y **AREA**).
- Enviar un formulario (**FORM**).
- Crear un documento con marcos (**FRAME** e **IFRAME**).
- Citar una referencia externa (**Q**, **BLOCKQUOTE**, **INS** y **DEL**).
- Hacer referencia a convenciones de metadatos que describen un documento (**HEAD**).

En general los URIs hacen distinción entre mayúsculas y minúsculas (es decir, que no es lo mismo `a` que `A`), por lo tanto es conveniente utilizar siempre letras minúsculas para las mismas.

Considerando el URI `http://www.vykus.com.ar/tfg`, podemos ver que estos están normalmente compuestos de la siguiente manera:

1. El esquema de nombres (protocolo) usado para acceder al recurso (Ej. `http`, `ftp` y `mailto`)
2. El nombre de la máquina que aloja el recurso (Ej. `www.vykus.com.ar`)
3. El nombre en sí del recurso, dado en forma de "path" o "ruta de acceso". (Ej. `/tfq`)

Para enviar un correo electrónico a un usuario se utilizaría el siguiente URI:

```
Consultas a <a href="mailto:info@vykus.com.ar">Maira Vykus</a>.
```

Hay que tener en cuenta que el término URL (Uniform Resource Locator) es un subconjunto dentro de los Identificadores Universales de Recursos.

Los URIs no aceptan valores no ASCII, sin embargo los autores los especifican en valores de atributos que esperan URIs (es decir, definidos con %URI; en el DTD) lo que los convierte en ejemplos **ilegales**:

```
<a href="http://www.dominio.com/Håkon">...</a>
```

Identificadores de fragmento

En la explicación anterior se podría agregar una cuarta parte en un URI, que hace referencia a una ubicación dentro de un recurso. Este tipo de URIs termina con un símbolo numeral (`#`) seguido de un identificador de vínculo o **fragmento**. Por ejemplo, el siguiente vínculo apunta a la segunda parte desarrollada en un documento:

```
http://www.vykus.com.ar/tfg/index.html#segundaparte
```

URIs relativos

Un **URI relativo** es aquel en el que no se especifica el esquema de nombres o máquina, sino que se coloca directamente el **nombre del recurso**, debido a que normalmente su ruta hace referencia a un recurso que está en la misma máquina que el documento actual.

Los URIs relativos pueden contener:

- indicadores relativos de ruta (Ej. `..` significa un nivel superior en la jerarquía de acceso)
- e identificadores de fragmento.

Los URIs relativos se convierten en **URIs completos o absolutos** a partir de un URI base. Por lo tanto, el siguiente URI relativo:

```
<a href="xhtml.html">XHTML</a>
```

Sería convertido de la siguiente manera:

```
<a href="http://www.vykus.com.ar/tfg/xhtml.html">XHTML</a>
```

Si tuviéramos el URI base `http://www.vykus.com.ar/tfg/index.html`.

Por otra parte si tuviéramos un vínculo a una imagen como el siguiente:

```

```

Se expandiría al URI completo "http://www.vykus.com.ar/imagenes/soporte.jpg" porque los dos puntos (..) le indican que debe bajar un nivel en la jerarquía.

ELEMENTO A (VÍNCULOS EN BODY)

Sintaxis: (etiqueta inicial y final obligatoria)

```
<a>...</a>
```

El elemento **A** (llamado como etiqueta de ancla) se utiliza comúnmente con el atributo **href** para crear un vínculo y sólo puede aparecer en el cuerpo (**BODY**) del documento. Al clickear con el mouse o seleccionar con el teclado dicho vínculo, se puede acceder a otra página en el mismo sitio o en uno diferente. Por defecto, la nueva página se carga normalmente en la misma ventana, a menos que se especifique lo contrario con el atributo **target**.

Entre las etiquetas de apertura y cierre de **A**, se pueden insertar caracteres, imágenes, quiebres de línea (**BR**) y texto, sin embargo no deben colocar otras etiquetas HTML o código de Hojas de Estilo en Cascada (esto debe estar fuera del elemento **A**).

La representación de los vínculos depende del agente de usuario o de la aplicación de hojas de estilos, sin embargo, los mismos suelen mostrarse de una forma obvia para los usuarios (subrayados, con los colores invertidos, etc.). La representación también puede variar si el usuario ya ha visitado el vínculo.

El agente de usuario puede manipular el recurso obtenido de diferentes maneras: abriendo un nuevo documento HTML en la misma ventana del agente de usuario, abriendo un nuevo documento HTML en una ventana diferente, iniciando un nuevo programa que maneje el recurso, etc.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de foco: onfocus, onblur de teclado: accesskey, tabindex mapas de imágenes: shape ("rect" "circle" "poly" "default"), coords (Coords) internacionalización: hreflang (LanguageCode), charset (Charset) otros: type (ContentType), name (NMTOKEN), href (URI), rel (LinkTypes), rev (LinkTypes)	target (FrameTarget)	Inline (acepta elementos en línea)	A (un elemento A no puede contener a otro A)
Desarrollo de atributos			
name name="un_nombre"	Asigna un nombre único al elemento actual para que actúe como destino de otro vínculo. El campo de acción de este nombre es el documento actual y se comparte el espacio de nombres con el atributo id .		
href href="http://dominio.com"	Este atributo se utiliza para especificar la dirección URL destino para un vínculo, definiendo así un vínculo entre el elemento actual (el origen del vínculo) y el destino del vínculo definido por este atributo.		

type <i>type="text/html"</i>	Indica el tipo de contenido disponible en el destino del vínculo. Permite a los agentes de usuario no obtener dicho contenido si no lo soportan. Los autores deben afrontar el riesgo que el valor asignado se inconsistente con el contenido del destino. Algunos tipos disponibles son: <code>image/jpeg</code> , <code>image/gif</code> , <code>image/png</code> , <code>image/tiff</code> , <code>text/css</code> , <code>text/html</code> , <code>application/postscript</code> , etc. (ver <code>contenttype</code>)
rel <i>rel="copyright"</i>	Es un tipo de vínculo directo aplicable en A y LINK que describe la relación entre el documento actual y el destino del vínculo (especificado por <code>href</code>). El valor de este atributo es una lista de tipos de vínculo separados por espacios. Algunos de los valores (ver <code>contenttype</code>) pueden ser: <code>appendix</code> , <code>bookmark</code> , <code>chapter</code> , <code>contents</code> , <code>copyright</code> , <code>glossary</code> , <code>help</code> , <code>index</code> , <code>next</code> , <code>prev</code> , <code>section</code> , <code>stylesheets</code> , y <code>subsection</code> . No está completamente soportado por muchos navegadores.
rev <i>rev="relation"</i>	Es un atributo aplicable en A y LINK que se utiliza para describir un vínculo inverso desde el origen del vínculo, (especificado por <code>href</code>) y el documento actual. El valor de este atributo es una lista de tipos de vínculo separados por espacios. El valor más útil es <code>relation</code> . No está completamente soportado por muchos navegadores.
hreflang <i>hreflang="en"</i>	Especifica el idioma del recurso designado por <code>href</code> y sólo puede utilizarse si se especifica este último. Proporciona información sobre el idioma del contenido o de los valores de los atributos de un elemento.
charset <i>charset="ISO-8859-1"</i>	Especifica la codificación de caracteres (ver <code>Codificación de caracteres</code>) de la página destino de un vínculo: Los agentes de usuario, provistos de esta información adicional, no deberían mostrar información "basura" al usuario. Para ello pueden localizar los recursos necesarios para la correcta presentación del documento o advertir al usuario que el documento será ilegible.

Cada elemento **A** define un origen o destino de vínculo:

1. El contenido de un elemento **A** define la posición del origen o del destino del vínculo.
2. Si el atributo `name` o el atributo `id` toman algún valor, el elemento define un posible destino de uno o más vínculos.
3. Si el atributo `href` tiene algún valor, el elemento define el origen de un vínculo que puede ser activado por el usuario para obtener un recurso de la Web. En dicho caso, el origen del vínculo está en el lugar donde aparece el elemento **A** (el texto Clarín) y el destino del vínculo es el recurso web (`http://www.clarin.com.ar`).

```
<p>Para más información visite <a href="http://www.clarin.com.ar">Clarín</a></p>
```
4. Los autores pueden establecer los atributos `name` y `href` al elemento a simultáneamente, con lo cuál dicho elemento es a la vez destino y origen.

```
<p>Como puede observarse en segundo párrafo del <a name="himno" href="segundoparrafo.html">Himno Nacional Argentino</a></p>
```
5. Los autores también pueden crear un elemento **A** que no sea origen ni destino de vínculo, es decir, que no especifique ni `href`, ni `name`, ni `id`. Los valores de estos atributos pueden ser establecidos posteriormente a través de scripts.

Nombre y destino de vínculos

El destino de un vínculo puede ser un elemento contenido dentro de un documento HTML. Dicho destino debe tener un nombre, que puede especificarse mediante:

- el atributo `name` del elemento **A**
- el atributo `id` de cualquier elemento (`h1`, `h2`, `p`, etc.)

Cualquier URI que se refiera a ese destino debe incluir el nombre como identificador de fragmento, es decir, con el signo # precediendo al nombre asignado al destino mediante el atributo **name** o **id**.

Los nombres de los vínculos deberían restringirse a los caracteres ASCII y deben seguir las siguientes reglas:

- **Unicidad:** Los nombres de vínculo deben ser únicos dentro de un documento y no pueden aparecer en un mismo documento nombres que sólo se diferencien por las letras mayúsculas y minúsculas. Ej. un elemento de nombre "mivinculo" y otro "MiVinculo" no pueden aparecer en el mismo documento.
- **Emparejamiento de cadenas:** Las comparaciones entre identificadores de fragmento y nombres de vínculos deben dar resultados exactos (incluyendo mayúsculas y minúsculas). Ej. si el identificador de fragmento es #mivinculo, entonces el nombre deberá ser **name="mivinculo"** y no **name="MiVinculo"**

Así, un ejemplo de nombres de vínculos correctos en ambos aspectos sería:

```
<p>Para más fotografías vea el título <a href="#mivinculo">Mis fotos</a></p>
...
<h2><a name="mivinculo">Mis fotos</a></h2>
...
```

Los atributos **id** y **name** comparten el mismo espacio de nombres, por lo que no pueden definir cada uno un destino de vínculo con un mismo nombre en el mismo documento.

Cuando **name** y **id** declaran el mismo nombre (a1) dos veces en un mismo documento, el código es inválido tal como el siguiente ejemplo:

```
<h1 id="a1">
<a name="a1"></a>
```

Sin embargo, está permitido usar ambos atributos para especificar un identificador único para un mismo elemento en los siguientes elementos: **A**, **FORM**, **FRAME**, **IFRAME**, **IMG** y **MAP**. En dicho caso, sus valores deben ser idénticos.

```
<p><a name="a1" id="a1">Un vínculo</a></p>
```

Al elegir entre el uso de **id** o **name** para la asignación de nombres se debe tener en cuenta que:

- El atributo **id** puede actuar como algo más que un nombre de vínculo (Ej. selector en una hoja de estilo, identificador en un proceso, etc.).
- Algunos agentes de usuario antiguos no soportan los vínculos creados con el atributo **id**.
- El atributo **name** permite nombres de vínculos más ricos (con entidades).

Ejemplos

Vínculo con title

CÓDIGO

```
<p>Para más información al respecto visite <a href="http://www.clarin.com.ar"
title="Sitio web diario digital Clarin.com.ar">Clarín</a>.</p>
```

Vínculo con rel y rev

Si aplicamos los atributos **rel** y **rev**, en los documentos A y B, los ejemplos significan exactamente lo mismo:

CÓDIGO DOCUMENTO A

```
<link href="docB" rel="blabla" />
```

CÓDIGO DOCUMENTO B

```
<link href="docA" rev="blabla" />
```

Vínculo con charset

CÓDIGO

```
Para más información sobre el W3C, consulte el
<a href="http://www.w3.org/" charset="ISO-8859-1">sitio web del W3C</a>
```

Destino de vínculo con atributo name

CÓDIGO: ESPECIFICACIÓN DEL DESTINO

Con el atributo **name** del elemento **A**, se puede asignar el nombre “segundotema” a un elemento **h2** del documento “prueba.html”; lo que generaría un destino de vínculo alrededor del texto “Segundo Tema”. (Cuando un elemento **A** solo define un destino de vínculo, no se representa de una manera especial)

```
<h2><a name="segundotema">Segundo Tema</a></h2>
```

CÓDIGO: VÍNCULO AL DESTINO

Luego, una vez definido el destino, se puede generar un vínculo hacia el mismo cuyo URI deberá contener el carácter “#” seguido del nombre del elemento (identificador de fragmento). El vínculo podrá ser:

- Relativo, en el mismo documento:

```
<a href="#segundotema">Segundo Tema</a>
```

- Relativo, en otro documento pero en el mismo directorio:

```
<a href="./prueba.html#segundotema">Segundo Tema</a>
```

- Absoluto:

```
<a href="http://www.sitio.com.ar/prueba.html#segundotema">Segundo Tema</a>
```

El atributo **name** puede contener referencias de caracteres. Así, el valor `Dürst` es válido para el atributo **name**, como lo es `Dül;rst`. El atributo **id**, por contra, no puede contener referencias de caracteres.

Nota: Algunos agentes de usuario pueden no encontrar el "destino-vacio" del siguiente fragmento HTML:

```
<a name="destino-vacio"></a>
<a href="#destino-vacio">Vínculo a destino vacío</a>
```

Destino de vínculo con atributo id

CÓDIGO: ESPECIFICACIÓN DEL DESTINO

En base al ejemplo anterior, se puede asignar el atributo **id="segundotema"** directamente al elemento **H2** sin necesidad de crear un elemento **A** con el atributo **name** dentro del mismo.

```
<h2 id="segundotema">Segundo Tema</h2>
```

CÓDIGO: VÍNCULO AL DESTINO

Ese destino es vinculado en otra parte del documento por medio del elemento **A**:

```
<p>Nota: Para más información, vea el <a href="#segundotema">Segundo Tema</a></p>
```

Nota: También se puede dar un nombre a un destino de vínculo en el elemento **A** con el atributo **id**, en lugar del atributo **name**:

```
<p>A continuación un párrafo del <a id="himno">Himno Nacional Argentino</a></p>
```

ELEMENTO LINK (VÍNCULOS EN HEAD)

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<link />
```

El elemento **LINK** es utilizado para establecer una relación entre el documento actual y otro recurso. Por no tener contenido, **no** son mostrados con el cuerpo del documento, sin embargo, la información relacional puede ser representada por los agentes de usuario como herramientas de navegación u otras formas. Este elemento puede ser utilizado más de una vez, y cada aparición debe estar dentro del elemento **HEAD**. Siempre se requiere un valor para el atributo **href**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout internacionalización: hreflang (LanguageCode), charset (Charset) otros: type (ContentType), href (URI), rel (LinkTypes), rev (LinkTypes), media (MediaDesc)	target (FrameTarget)	Empty	
Desarrollo de atributos			
media media="aural "	El atributo media especifica a que medios se aplicará el vínculo. Los valores permitidos (ver MediaDesc) son: all, aural, braille, handheld, print, projection, screen, tty y tv.		
type type="text/html"	Especifica el tipo MIME (Multipurpose Internet Mail Extension) del documento vinculado. Algunos tipos (ver contenttype) disponibles son: image/jpeg, image/gif, image/png, image/tiff, text/css, text/html, application/postscript, etc.		

Ejemplo

Pueden aparecer varias definiciones link en la sección **HEAD** de un documento, con diferentes objetivos o funciones cada uno.

1. **Relaciones del documento vinculado con el documento actual**, a través de los atributos **rel** y **rev** con sus diferentes tipos de relaciones. En el ejemplo, en el encabezado del documento "Sección 8" se describe la posición del mismo dentro de una serie de documentos, indicando cual es la tabla de contenidos y cuál es la sección anterior y posterior:

```
<head>
<title>Sección 8</title>
<link rel="index" href="../index.html" />
<link rel="prev" href="seccion7.html" />
<link rel="next" href="seccion9.html" />
</head>
```

2. **Vínculos con hojas de estilo externas.** En ellos, el atributo **type** especifica el lenguaje de la hoja de estilo, y el atributo **media** especifica el medio o medios de representación deseados. Los agentes de usuario pueden ahorrar tiempo obteniendo de la red sólo aquellas hojas de estilo que se apliquen al dispositivo actual.

```
<link media="screen" type="text/css" href="../principal.css" />
```

3. **Vínculos a versiones alternativas de un documento, escritas en otro idioma.** Se puede utilizar:

- **hreflang**: para decir dónde encontrar la versión en árabe de un documento.
- **charset**: para indicar la codificación de caracteres.
- **title**: para describir el contenido.
- **type**: para indicar el tipo de contenido.
- **rel**: para indicar que es una versión alternativa.
- **href**: para indicar la ubicación del recurso.

```
<link title="Artículo en Árabe"
      type="text/html"
      rel="alternate"
      charset="ISO-8859-6"
      hreflang="ar"
      href="http://www.clarin.com.ar/idiomas/arabe/543.html" />
```

4. Vínculos a versiones alternativas de un documento, diseñadas para medios diferentes (como impresión).

```
<link media="print" title="Artículo para impresión"
      type="application/postscript"
      rel="alternate"
      href="http://www.clarin.com.ar/impresion/prueba.html" />
```

5. Vínculos a la página inicial de un conjunto de documentos.

```
<link rel="start" title="Página de inicio del tutorial"
      type="text/html"
      href="http://www.clarin.com.ar/tutorial/inicio.html" />
```

Los últimos tres son utilizados para proporcionar distintas informaciones a los motores de búsqueda de modo que estos mejoren el procesamiento de los documentos.

ELEMENTO BASE (RUTA DE ACCESO)

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<base href="URI" />
```

El elemento **BASE** es utilizado para definir la URL base para el documento HTML. Esto permite el uso de URL relativas en el documento. Esta URL base no necesariamente tiene que ser absoluta.

Además, cuando se accede al sitio en cuestión desde otro sitio mediante un link, se puede utilizar el atributo **target** del elemento base para asegurarse que el sitio no aparezca dentro de un frame del sitio anterior, sino siempre en primer plano.

Si está presente, el elemento **BASE** debe aparecer en la sección **HEAD** del documento HTML, antes de cualquier elemento que haga referencia a una fuente externa. La información de ruta especificada por el elemento **BASE** sólo afecta a los URIs del documento en el cual aparece el elemento.

Los URIs relativos se completan de acuerdo con un URI base, el cual puede provenir de distintas fuentes:

- un URI base por defecto, que es el del documento actual. Hay que tener en cuenta que en un mensaje de correo electrónico con HTML no existe un URI base, por lo tanto hay que asignarlo de manera explícita.
- un elemento **BASE** que permite a los autores especificar el URI base de un documento explícitamente.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id otros: href* (URI)	target (FrameTarget)	Empty	

Desarrollo de atributos	
href <i>href="http://www.dominio.com/"</i>	Este atributo obligatorio especifica un URI absoluto que actúa como el URI base para completar URIs relativos.
target <i>type="text/html"</i>	Información sobre el marco destino.

Ejemplo

CÓDIGO

Por ejemplo, dadas las siguientes declaraciones **BASE** y **A**:

```
<!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="en-US" lang="en-US">
<head>
  <title>Deportes</title>
  <base href="http://www.clarin.com.ar/deportes/inicio.html" />
</head>

<body>
  <p>Visite la sección <a href="../espectaculos/inicio.html">Espectáculos</a></p>
</body>
</html>
```

el URI relativo "../espectaculos/inicio.html" se completaría a:

```
http://www.clarin.com.ar/espectaculos/inicio.htm
```

9. Objetos, imágenes y aplicaciones



INTRODUCCIÓN A LOS OBJETOS, IMÁGENES Y APLICACIONES

XHTML permite incluir imágenes, aplicaciones (programas que se descargan automáticamente y se ejecutan en la máquina del usuario), videoclips, y otros documentos HTML en sus páginas.

El elemento **OBJECT** ofrece una solución universal para la inclusión de objetos genéricos e indica la información necesaria para la representación de un objeto: código fuente, valores iniciales y datos en tiempo de ejecución. Este elemento asume las tareas realizadas por los elementos existentes y sirve como solución de implementación de tipos de medios futuros.

Así, cada tipo de inclusión tiene una solución específica y una general:

- Para incluir imágenes, se puede usar el elemento **OBJECT** o el elemento **IMG**.
- Para incluir aplicaciones, se debe usar el elemento **OBJECT** ya que el elemento **APPLET** está desaprobadado.
- Para incluir un documento HTML en otros, se puede usar el elemento **IFRAME** o el elemento **OBJECT**. En ambos casos, el documento incluido sigue siendo independiente del documento principal. Los agentes de usuario visuales pueden presentar el documento incluido en una ventana diferenciada del documento principal.

En forma esquemática lo expondríamos de la siguiente manera:

Inclusión de:	Elemento específico	Elemento genérico
Imagen	IMG	OBJECT
Aplicación	APPLET (Desaprobado)	OBJECT
Otro documento HTML	IFRAME	OBJECT

Los objetos incluidos pueden tener hipervínculos asociados a ellos, tanto a través de los mecanismos de vinculación estándar, como también a través de mapas de imágenes. Un **mapa de imágenes** especifica las regiones geométricas activas de un objeto incluido, y asigna un vínculo a cada región. Cuando se activan, estos vínculos pueden hacer que se abra un documento, que se ejecute un programa en el servidor, etc.

ELEMENTO IMG

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<img />
```

El elemento **IMG** se utiliza para insertar una imagen (gráfico, fotografía, etc.) directamente en el flujo del texto y otras imágenes. No se insertan quiebres de línea o retornos de carro antes o después de este elemento. Se pueden utilizar estilos para afectar la apariencia y presentación de las imágenes.

Colocado dentro de un elemento **A** provee una imagen con vínculo, en cuyo caso, aparece un borde rodeando la imagen vinculada, que puede ser removido mediante hojas de estilos.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout mapas de imágenes: usemap (URI), ismap (ismap) otros: src* (URI), alt* (Text), longdesc (URI), height (Length), width (Length),	align (ImgAlign), border (Pixels), hspace (Pixels), vspace (Pixels), name (NMTOKEN)	Empty	
Desarrollo de atributos			
alt alt="cadena detexto"	Especifica texto alternativo (obligatorio en IMG , AREA y opcional en INPUT) para agentes de usuario que no puedan mostrar imágenes, formularios o aplicaciones. El idioma de este texto alternativo está especificado por el atributo lang . Agregar texto alternativo ayuda a los usuarios: <ul style="list-style-type: none"> • que no tengan terminales gráficas, • cuyos navegadores no soporten formularios • con discapacidades visuales • que utilicen sintetizadores de voz • que hayan configurado sus agentes de usuario para no mostrar imágenes, etc. Se deben seguir las siguientes pautas: <ul style="list-style-type: none"> • No especificar texto alternativo irrelevante cuando las imágenes sólo sirven para dar formato a una página. En tales casos, el texto alternativo debería ser la cadena vacía (""), aunque se aconseja evitar el uso de imágenes para dar formato a las páginas (utilizar hojas de estilo en su lugar). Por ejemplo, alt="flores" es inapropiado para una imagen que coloca una flor delante de un párrafo. • No especificar texto alternativo sin significado (Ej. "relleno"). Esto no solamente frustrará a los usuarios, sino que ralentizará a los agentes de usuario que deban convertir texto a salida por voz o Braille. 		

Atributos aceptados	Modelo de contenido
src <i>src="URI"</i>	Este atributo obligatorio indica la localización del recurso de imagen a través de un URI. El URI debe incluir el nombre de la imagen y una ruta absoluta o relativa. Algunos formatos de imagen reconocidos son: .gif, .jpg y .png
longdesc <i>longdesc="descripcionlarga"</i>	Este atributo especifica un vínculo a una descripción larga de la imagen, la cuál debería ser un complemento de la descripción corta proporcionada mediante al atributo alt . Cuando la imagen tiene asociado un mapa de imágenes, este atributo debería proporcionar información sobre los contenidos del mapa de imágenes.
width <i>width="80px"</i>	Dice a los agentes de usuario que invaliden el ancho original de la imagen u objeto en favor de este valor. Da una idea del tamaño de una imagen u objeto para que los agentes de usuario puedan reservar espacio y continuar la representación del documento mientras esperan a los datos de la imagen. Cuando el objeto es una imagen, se escala. Las longitudes expresadas como porcentajes se basan en el espacio horizontal disponible actualmente, y no en el tamaño original de la imagen, objeto o aplicación.
height <i>height="50px"</i>	Dice a los agentes de usuario que invaliden el alto original de la imagen u objeto en favor de este valor. Las longitudes expresadas como porcentajes se basan en el espacio vertical disponible actualmente, y no en el tamaño original de la imagen, objeto o aplicación.
name <i>name="unnombre"</i>	Este atributo asigna un nombre al elemento de modo que se pueda hacer referencia a él desde hojas de estilo o scripts. Nota: Este atributo ha sido incluido por motivos de compatibilidad con versiones anteriores. Las aplicaciones deberían usar el atributo id para identificar a los elementos.
usemap	Este atributo asocia un mapa de imágenes con un elemento. El mapa de imágenes está definido por un elemento MAP . El valor de usemap debe ser igual al valor del atributo name del elemento MAP asociado.

Ejemplo

INSERTAR IMAGEN CON IMG U OBJETO

Una foto insertada con el elemento **IMG** resultaría en:

```

```

La misma foto se insertaría con el elemento **OBJECT** de la siguiente manera:

```
<object data="evento.gif" type="image/png">Foto de apertura evento 2007.</object>
```

IMAGEN CON ALT Y LONGDESC

A través de la descripción corta proporcionada por **alt**, el usuario podrá decidir si desea ver la descripción más larga declarada por **longdesc** en el archivo atomo.html.

```

```

ELEMENTO OBJECT

Sintaxis: (etiqueta inicial y final obligatoria)

```
<object>...</object>
```

El elemento **OBJECT** es utilizado para insertar un objeto (componentes ActiveX, applets, mapas de imágenes, media

players, y plug-ins) dentro de un documento HTML y proveer toda la información necesaria para implementar y correr dicho objeto.

Permite la opción de declarar e iniciar el objeto en el mismo momento, o declarar el objeto e iniciar una o más veces luego. Se puede insertar texto entre las etiquetas de apertura y cierre, que será mostrado como el mensaje por defecto si el objeto no puede ser mostrado por el navegador.

Este elemento puede ubicarse dentro del elemento **HEAD** o **BODY**. Si está dentro del **HEAD**, no debe incluir contenido para mostrar dentro de la página.

Acepta elementos **PARAM** y otros elementos en bloque y en línea. El elemento **PARAM** debe colocarse dentro del elemento **OBJECT** para brindar parámetros requeridos para implementar el objeto. Este elemento debe aparecer inmediatamente después de la apertura de **OBJECT**, pero antes de cualquier otro elemento.

Entonces, con el elemento **OBJECT** se especifica la implementación de un objeto y la localización de los datos a representar y con el elemento **PARAM**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout de teclado: tabindex (Number) mapas de imágenes: usemap (URI) otros: <u>declare</u> ("declare"), <u>classid</u> (URI), <u>codebase</u> (URI), <u>data</u> (URI), <u>type</u> (ContentType), <u>codetype</u> (ContentType), <u>archive</u> (UriList), <u>standby</u> (Text), <u>height</u> (Length), <u>width</u> (Length), <u>name</u> (NMTOKEN)	align (ImgAlign), border (Pixels), hspace (Pixels), vspace (Pixels)	(PARAM Flow)*	
Desarrollo de atributos			
classid <i>classid="URI"</i>	Este atributo puede utilizarse para especificar la localización de un objeto mediante un URI. Puede usarse junto con, o como alternativa al atributo data , dependiendo del tipo de objeto involucrado. Si la URL es relativa debe proveerse una URL base a través de codebase . Cuando se utiliza este atributo, también se puede especificar el tipo de código utilizando el atributo codetype .		
codebase <i>codebase="URI"</i>	Especifica la ruta de acceso base (base path) utilizado para completar los URIs relativos especificados por los atributos classid , data y archive . Si está ausente, su valor por defecto es el URI base del documento actual.		
codetype <i>codetype=image/gif""</i>	Especifica el tipo esperado de contenido de datos cuando se carga el objeto especificado por classid . Este atributo es opcional pero se recomienda cuando se especifica classid ya que permite que el agente de usuario evite la carga de información de tipos de contenido que no soporta. Cuando está ausente, su valor por defecto es el valor del atributo type .		
data <i>data="URI"</i>	Puede utilizarse para especificar la localización de los datos del objeto, por ejemplo datos de imágenes para objetos que definen imágenes, o más en general, una forma serializada de un objeto que puede usarse para recrearlo. Si se da como URI relativo, debería interpretarse relativamente al atributo codebase .		

Atributos aceptados	Modelo de contenido
type <i>type="image/gif"</i>	Especifica el tipo de contenido de los datos especificados por data . Este atributo es opcional, pero se recomienda incluirlo cuando se especifica data , ya que permite que el agente de usuario no cargue información de tipos de contenido que no soporta. Si el valor de este atributo es diferente del "Content-Type" HTTP devuelto por el servidor cuando se obtiene el objeto, el "Content-Type" HTTP tiene preferencia.
archive <i>archive="URI"</i>	Este atributo puede utilizarse para especificar una lista de URIs separados por espacios de archivos que contienen recursos relevantes para el objeto, los cuales pueden incluir los recursos especificados por los atributos classid y data . La precarga de archivos resultará generalmente en tiempos menores de descarga para los objetos. Los archivos especificados como URIs relativos deberían interpretarse relativamente al atributo codebase .
declare <i>declare="declare"</i>	Cuando está presente, este atributo booleano hace que la definición actual de object sea solamente una declaración, previniendo que los navegadores bajen y ejecuten el objeto. Al declarar el objeto se le debe asignar un valor al atributo id , que luego será referenciado al crear el objeto.
standby <i>standby="mensaje"</i>	Este atributo especifica un mensaje que puede presentar un agente de usuario mientras carga la implementación y los datos del objeto.

La mayoría de los agentes de usuario tienen mecanismos para representar tipos de datos comunes como texto, imágenes GIF, colores, fuentes, etc. Sin embargo, para representar tipos de datos para los cuales no tienen soporte suelen ejecutar aplicaciones externas. El elemento **OBJECT** permite a los autores especificar si los datos deben ser representados externamente por algún programa dentro del agente de usuario. En ese caso el autor debería especificar:

- **La implementación del objeto incluido.** Por ejemplo, si el objeto incluido es una aplicación de reloj, el autor debe indicar la localización del código ejecutable de la aplicación.
- **Los datos que deben representarse.** Por ejemplo, si el objeto incluido es un programa que representa datos de fuentes, el autor debe indicar la localización de estos datos.
- **Otros valores que necesite el objeto en tiempo de ejecución.** Por ejemplo, algunas aplicaciones pueden requerir valores iniciales para los parámetros.

El elemento **OBJECT** permite especificar estos tres tipos de datos, pero los autores no tienen que especificar necesariamente los tres a la vez, ya que algunos objetos pueden no necesitar información adicional sobre la implementación, como en el caso de las imágenes gif en que el agente de usuario ya sabe como representarlas.

APLICACIONES

Uso de texto alternativo

Si a través del elemento **OBJECT** insertamos una aplicación en python, no necesitaremos datos ni valores en tiempo de ejecución, sólo habrá que aclarar el atributo **classid** que especifica la localización de la aplicación:

```
<object classid="reloj.py">Un reloj animado.</object>
```

Es recomendable completar la declaración con un texto alternativo como contenido de **OBJECT** por si el agente de usuario no puede representar el reloj.

Objetos anidados

El elemento **OBJECT** ofrece un mecanismo para especificar representaciones alternativas del objeto, ya que cada declaración **OBJECT** anidada puede especificar tipos de contenido alternativos. Si un agente de usuario no puede

representar el primer **OBJECT**, intentará representar los contenidos, que a su vez pueden ser otro elemento **OBJECT**.

Anidamos varias declaraciones **OBJECT** para ilustrar el funcionamiento de las representaciones alternativas. El agente de usuario intentará representar el primer elemento **OBJECT** que pueda, en el siguiente orden: (1) una animación MPEG de un tsunami, (2) una imagen GIF del tsunami, (2) texto alternativo.

```
<object data="tsunami.mpeg" type="application/mpeg">
  <object data="tsunami.gif" type="image/gif">
    <strong>Tsunami</strong> visto desde el aire.
  </object>
</object>
```

El texto alternativo se proporciona por si todos los mecanismos anteriores fallaran.

Esquemas globales de nombres para objetos

La localización de la implementación de un objeto viene dada por un URI. Si bien el esquema por defecto es http, existen otros esquemas: los URIs comienzan con **java** al especificar una aplicación Java y con **clsid** para aplicaciones ActiveX.

CÓDIGO APLICACIÓN JAVA

```
<object codetype="application/java-archive"
  classid="java:program.start">
</object>
```

Con el atributo **codetype** los agentes de usuario pueden decidir no obtener la aplicación Java si no tienen la capacidad de soportarla.

Algunos esquemas de representación necesitan información adicional para identificar su implementación, y hay que decirles dónde encontrar esa información, con el atributo **codebase**:

```
<object codetype="application/java-archive"
  classid="java:program.start">
  codebase="http://www.dominio.com/implementacion/"
</object>
```

CÓDIGO APLICACIÓN ACTIVE X

Con el atributo **classid** se puede especificar un objeto ActiveX a través de un URI que comienza con el esquema de nombres "**clsid**". También se usa el atributo **data** que localiza los datos a representar:

```
<object classid="clsid:663C8FEF-1EF9-11CF-A3DB-080036F12502"
  data="http://www.dominio.com/reloj.stm">
  Esta aplicación no está soportada.
</object>
```

Declaración y creación de objetos

Cuando un documento contiene varios ejemplares del mismo objeto, es posible separar la declaración de un objeto de su creación, lo que implica varias ventajas:

- El agente de usuario puede obtener los datos de la red una vez (durante la declaración) y reutilizarlos para cada ejemplar.
- Es posible crear un objeto desde una localización diferente a la de la declaración del objeto, por ejemplo, desde un vínculo.
- Es posible especificar objetos como datos de ejecución de otros objetos.

Para declarar un objeto de modo que no sea ejecutado cuando lo lea el agente de usuario se establece:

- el atributo booleano **declare** del elemento **OBJECT**,

- el atributo **id** del elemento **OBJECT** es un valor único que identifica la declaración. Las creaciones posteriores del objeto se referirán a este identificador.

Un **OBJECT** declarado debe aparecer en el documento antes de la primera creación de ese **OBJECT**. Un objeto definido con el atributo **declare** se crea cada vez que un elemento que se refiera a ese objeto necesite que sea representado (Ej. se activa un vínculo que se refiere a él, se activa un objeto que se refiere a él, etc.).

Declaramos un **OBJECT** y hacemos que se cree haciendo referencia a él desde un vínculo, con lo cual el objeto puede activarse haciendo clic en un texto resaltado:

```
<object declare="declare"
      id="luna.declaracion"
      data="Luna.mpeg"
      type="application/mpeg">
  La Luna desde La Tierra.
</object>
...más abajo en el documento...
<p>¡Una <a href="#luna.declaracion"> animación de la Luna!</a></p>
```

Otra manera de crear un objeto declarado es:

```
<object data="#luna.declaracion"></object>
```

Inclusión de documentos

Para incluir un documento en otro documento (en lugar de crear un vínculo), se pueden utilizar los elementos **IFRAME** y **OBJECT**, que difieren en varios aspectos. Tienen modelos de contenido diferentes e **IFRAME** puede ser un marco destino y puede ser "seleccionado" por un agente de usuario dirigiendo a él el foco (para imprimir, ver el fuente HTML, etc.) Los agentes de usuario pueden representar los marcos seleccionados de modo que se distingan de los marcos no seleccionados (Ej. dibujando un borde alrededor del marco seleccionado).

Un documento sólo se representa dentro de otro documento, y en los demás aspectos es completamente independiente del documento en el cual se incluye. Por ejemplo, los URIs relativos del documento incluido se completan de acuerdo con el URI base del documento incluido, y no con del documento principal.

Por ejemplo, la línea siguiente incluye el contenido del archivo.html en donde aparece la definición de **OBJECT** (su contenido sólo debe representarse si no se puede cargar el archivo especificado por **data**):

```
...texto antes...
<object data="archivo.html">No se pudo incluir archivo.html</object>
...texto después...
```

ELEMENTO PARAM

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<param />
```

Los elementos **PARAM** especifican un conjunto de valores que pueden ser necesarios para un objeto en tiempo de ejecución, puede aparecer cualquier cantidad de estos elementos y en cualquier orden, inmediatamente luego de la apertura de un elemento **OBJECT**.

La etiqueta **PARAM** se utiliza para establecer una dupla **name/value** que provee los parámetros necesarios por un objeto. Cada elemento **PARAM** puede proveer sólo un parámetro, los cuáles, incluyendo el nombre y el rango aceptado de valores, son establecidos por el autor del objeto.

Atributos aceptados		Modelo de contenido	
XHTML Strict		Frameset, Transitional	Contiene a: No acepta:
principales: id otros: <u>name</u> (CDATA), <u>value</u> (CDATA), <u>valuetype</u> ("data" "ref" "object"), <u>type</u> (ContentType)			Empty
Desarrollo de atributos			
name <i>name="alto"</i>	Este atributo define el nombre de un parámetro de ejecución, que se supone que el objeto insertado conoce. Dependiendo de la implementación específica del objeto, se distinguirá o no entre mayúsculas y minúsculas.		
value <i>value="40"</i>	Este atributo especifica el valor del parámetro de ejecución especificado por name . Los valores de las propiedades no tienen significado para HTML; su significado lo determina el objeto en cuestión.		
valuetype <i>valuetype="data"</i> <i>valuetype="ref"</i> <i>valuetype="object"</i>	Este atributo especifica el tipo de atributo value y los valores posibles son: <ul style="list-style-type: none"> data: Este es el valor por defecto del atributo. Significa que el valor especificado por value se evaluará y pasará a la implementación del objeto como una cadena. ref: El valor especificado por value es un URI que designa un recurso donde se almacenan valores de tiempo de ejecución. Esto permite a las herramientas identificar los URIs dados como parámetros. El URI debe pasarse al objeto tal y como está, es decir, sin completar. object: El valor especificado por value es un identificador que se refiere a una declaración OBJECT del mismo documento. El identificador debe ser el valor del atributo id establecido para el objeto OBJECT declarado. 		
type <i>type="text/html"</i>	Este atributo especifica el tipo de contenido (MIME type) del recurso contenido por el atributo value sólo en el caso en que valuetype sea igual a "ref" . Este atributo especifica así para el agente usuario el tipo de valores que encontrará en el URI designado por value.		

Aplicaciones

Ancho y alto con param

En el ejemplo del reloj se puede utilizar param para indicar el ancho y alto de la siguiente manera:

```
<object classid="reloj.py">
  <param name="alto" value="40" valuetype="data" />
  <param name="ancho" value="40" valuetype="data" />
  Un reloj animado.</object>
```

Datos de ejecución con param

En otro ejemplo se puede especificar los datos de ejecución para el parámetro "Valores_inic" del objeto como recurso externo (un fichero GIF).

```
<object classid="http://www.dominio.com/aplicgif" standby="Cargando La Luna...">
  <param name="Valores_inic"
    value="./imagenes/luna.gif"
    valuetype="ref" />
</object>
```

Allí, el valor del atributo **valuetype** se establece en **"ref"**, **value** es un URI que designa el recurso y se designa el atributo **standby** para que el agente de usuario muestre un mensaje mientras se carga el mecanismo de

representación.

Param y object anidados

Al representar un elemento **OBJECT**, los agentes de usuario sólo deben buscar el contenido de los elementos **PARAM** que son hijos directos y dar dicho contenido al **OBJECT**.

Por esto, si se representa "objeto1" con su "param1" y "objeto2" con su "param2" dentro del primer objeto:

- "param1" sólo se aplica a "objeto1" y no a "objeto2".
- si se representa "objeto2" porque "objeto1" no pudo mostrarse, el "param1" no se tiene en cuenta y "param2" se aplica al "objeto2".
- si ninguno de los **OBJECT** es representado, entonces ningún **PARAM** se aplica.

```
<object id="objeto1">
  <param name="param1" />
  <object id="objeto2">
    <param name="param2" />
  </object>
</object>
```

MAPAS DE IMÁGENES

Los mapas de imágenes permiten especificar regiones en una imagen u objeto y asignar una acción específica a cada región (Ej. abrir un documento, ejecutar un programa, etc.), de modo que cuando la región sea activada por el usuario, la acción se ejecute.

Un mapa de imágenes se crea asociando un objeto con una especificación de las áreas geométricas sensibles del objeto.

Hay dos tipos de mapas de imágenes:

- **En el lado del cliente.** Cuando un usuario activa con el mouse una región de un mapa las coordenadas en píxeles son interpretadas por el agente de usuario. El agente de usuario selecciona el vínculo especificado por la región activada y lo sigue.
- **En el lado del servidor.** Cuando un usuario activa con el mouse una región de un mapa del lado del servidor, las coordenadas en píxeles son enviadas al agente del lado del servidor especificado por el atributo **href** del elemento **A**. El agente del servidor interpreta las coordenadas y realiza alguna acción.

Se prefieren los mapas de imágenes en el cliente que los mapas de imágenes en el servidor por dos razones: son accesibles a las personas que utilizan agentes de usuario no gráficos y permiten saber en todo momento si el apuntador está sobre una región activa o no.

Mapas de imágenes del lado del cliente

Elemento MAP

Sintaxis: (etiqueta inicial y final obligatoria)

```
<map> . . . </map>
```

El elemento **MAP** especifica un mapa de imágenes en el lado del cliente dividido en dos o más regiones sensibles al mouse, que puede ser asociado con otros elementos (**IMG**, **OBJECT** o **INPUT**) a través del atributo **usemap** del elemento. El elemento **MAP** puede ser utilizado sin una imagen asociada por mecanismos generales de navegación. Por ejemplo, al hacer clic en una región específica del mapa de imágenes, un vínculo puede enviarnos a otra web.

La presencia del atributo **usemap** en un elemento **OBJECT** implica que el objeto que incluye es una imagen. Cuando el elemento **OBJECT** tiene asociado un mapa de imágenes en el lado del cliente, los agentes de usuario pueden permitir la interacción del usuario con el elemento **OBJECT** sólo en lo que se refiere al mapa de imágenes en el lado del cliente. Esto permite a los agentes de usuario (tales como un navegador de audio o un robot) interactuar con el **OBJECT** sin tener que procesarlo. Cuando un **OBJECT** tiene asociado un mapa de imágenes, el objeto no siempre es obtenido o procesado por todos los agentes de usuario.

El modelo de contenido del elemento **MAP** permite:

1. Uno o más elementos **AREA**, que no tienen contenido, sino que especifican las regiones geométricas del mapa de imágenes y los vínculos asociados con cada región. Los agentes de usuario generalmente no representan los elementos **AREA**, por lo tanto hay que proporcionar texto alternativo para cada una con el atributo.
2. Contenido en bloque, que debería incluir elementos **A** que especifiquen regiones geométricas del mapa de imágenes y el vínculo asociado con cada región. Obsérvese que el agente de usuario debería representar el contenido en bloque del elemento **MAP**.

Cuando un elemento **MAP** está formado por contenido mixto (tanto elementos **AREA** como contenido en bloque), los agentes de usuario deberían ignorar los elementos **AREA**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id*, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout otro: <u>name</u> (NMTOKEN)		(Block+ AREA +)	
Desarrollo de atributos			
name name="nombre"	Asigna un nombre al mapa de imágenes definido por un elemento map . El nombre debe ser único y distingue entre mayúsculas y minúsculas.		

Elemento AREA

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<area />
```

El elemento **AREA** es usado (junto con el elemento **MAP**) para crear un mapa de imágenes en el lado del cliente dividido en dos o más regiones sensibles al mouse. Al hacer clic en una región del mapa, un atributo contenido en el elemento **AREA** puede hacer que ocurra una acción. Si se crea un vínculo con el atributo **href**, al hacer clic en dicha región, el vínculo nos llevará a otra página. Este elemento debe estar contenido dentro del elemento **MAP**.

Los autores deberían especificar la geometría de un mapa de imágenes completamente con elementos **AREA**, completamente con elementos **A**, o completamente con ambos si el contenido es mixto. Los autores pueden querer usar contenido mixto para que los agentes de usuario antiguos utilicen la geometría del mapa especificada por los elementos **AREA** y que los agentes de usuario modernos saquen partido de la riqueza del contenido en bloque.

Si dos o más regiones se superponen, tiene prioridad la región definida por el elemento que aparece antes en el documento.

Los agentes de usuario y los autores deberían ofrecer alternativas textuales a los mapas de imágenes gráficos para los casos en que los gráficos no estén disponibles o en que el usuario no pueda acceder a ellos. Por ejemplo, los agentes de usuario pueden usar el texto **alt** para crear vínculos textuales en lugar de un mapa de imágenes gráfico. Estos vínculos pueden ser activados de diferentes maneras (con el teclado, activación por voz, etc.).

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id*, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de foco: onfocus, onblur. de teclado: acceskey, tabindex otro: <u>shape</u> ("rect" "circle" "poly" "default"), <u>coords</u> (Coords), <u>href</u> (URI), <u>nohref</u> ("nohref") <u>alt</u> * (Text)	target (FrameTarget)	Empty	
Desarrollo de atributos			
shape shape="default" shape="rect" shape="circle" shape="poly"	Este atributo especifica la forma de una región. No está soportado por muchos navegadores. Sus valores posibles son: <ul style="list-style-type: none"> • default: Especifica la región completa. • rect: Define una región rectangular. • circle: Define una región circular. • poly: Define una región poligonal. 		
coords coords="rect" coords="circle" coords="poly"	Este atributo especifica la posición y la forma en la pantalla, definiendo un área de influencia alrededor del elemento A . El número y orden de estos valores depende de la forma que está siendo definida. No está soportado por muchos navegadores. Combinaciones posibles: <ul style="list-style-type: none"> • rect: x-izquierda, y-superior, x-derecha, y-inferior. • circle: x-centro, y-centro, radio. Nota: Cuando el valor del radio sea un valor porcentual, los agentes de usuario deberían calcular el valor final del radio basándose en el ancho y alto del objeto asociado. El radio debería ser el menor valor de los dos. • poly: x1, y1, x2, y2, ..., xN, yN. El primer par de coordenadas x, y debería ser igual al último para cerrar el polígono. Cuando estas coordenadas no sean iguales, los agentes de usuario deberían añadir un par adicional de coordenadas para cerrar el polígono. Las coordenadas son relativas a la esquina superior izquierda del objeto, los valores son longitudes y van separados por comas. 		
nohref nohref="nohref"	Este atributo booleano especifica que una región no tiene asociado ningún vínculo.		

Ejemplos

Map dentro de Object

Si queremos que los contenidos del elemento **MAP** sólo sean representados cuando el **OBJECT** no puede serlo, "ocultamos" el elemento **MAP** dentro del contenido del elemento **OBJECT**. Así, cuando el elemento **OBJECT** puede ser mostrado, el elemento **MAP** no lo será:

```
<object data="navegacion.gif" type="image/gif" usemap="#mapal">
  <map name="mapal">
    <p>Navegar por este sitio:</p>
    <a href="guia.html" shape="rect" coords="0,0,118,28">Acceder a la Guía</a> |
    <a href="atajo.html" shape="rect" coords="118,0,184,28">Ir</a> |
    <a href="buscar.html" shape="circle" coords="184,200,60">Buscar</a> |
    <a href="top10.html" shape="poly"
      coords="
276,0,276,28,100,200,50,50,276,0">Top Ten</a>
  </map>
```

```
</object>
```

Map fuera de Object

En otros casos, se puede necesitar que el contenido del mapa se represente incluso si el agente de usuario puede representar el **OBJECT**. Para ello, definimos el elemento **MAP** fuera del **OBJECT**:

```
<object data="navegacion.gif" type="image/gif" usemap="#mapa1"></object>

... más contenido...

<map name="mapa1">
  <p>Navegar por este sitio:</p>
  <a href="guia.html" shape="rect" coords="0,0,118,28">Acceder a la Guía</a> |
  <a href="atajo.html" shape="rect" coords="118,0,184,28">Ir</a> |
  <a href="busca.html" shape="circle" coords="184,200,60">Buscar</a> |
  <a href="top10.html" shape="poly"
    coords="276,0,276,28,100,200,50,50,276,0">Top Ten</a>
</map>
```

Map con Area

Un mapa de imágenes similar creado utilizando el elemento **AREA** con el atributo **alt**:

```
<object data="navegacion.gif" type="image/gif" usemap="#mapa1">Esto es una barra
de navegación.</object>

<map name="mapa1">
  <area href="guia.html"
    alt="Acceder a la Guía"
    shape="rect"
    coords="0,0,118,28" />
  <area href="busca.html"
    alt="Buscar"
    shape="rect"
    coords="184,0,276,28" />
  <area href="atajo.html"
    alt="Ir"
    shape="circle"
    coords="184,200,60" />
  <area href="top10.html"
    alt="Top Ten"
    shape="poly"
    coords="276,0,276,28,100,200,50,50,276,0" />
</map>
```

Esto mismo se puede hacer usando el elemento **IMG** en lugar de **OBJECT**:

```

```

Object anidado con Map

Los mapas de imágenes pueden compartirse. Los elementos **OBJECT** anidados son útiles para proporcionar "redes de seguridad" en caso de que un agente de usuario no soporte ciertos formatos.

Si el agente de usuario no soporta el formato PNG, intentará representar la imagen GIF y si no es capaz de usarla (un agente de usuario por voz), utilizará la descripción textual proporcionada como contenido del elemento **OBJECT** interior. Como se puede ver en el ejemplo, los dos **OBJECTS** comparten el elemento mapa:

```
<object data="navegacion.png" type="image/png" usemap="#mapa1">
  <object data="navegacion.gif" type="image/gif" usemap="#mapa1">
    <map name="mapa1">
      <p>Navegar por este sitio:</p>
      <a href="guia.html" shape="rect" coords="0,0,118,28">Acceder a la Guía</a> |
```

```

<a href="atajo.html" shape="rect" coords="118,0,184,28">Ir</a> |
<a href="buscar.html" shape="circle" coords="184,200,60">Buscar</a> |
<a href="top10.html" shape="poly"
  coords="276,0,276,28,100,200,50,50,276,0">Top Ten</a>
</map>
</object>
</object>

```

Map con A

Pueden especificarse elementos **A** para crear zonas inactivas dentro de un mapa de imágenes.

El primer vínculo especifica una pequeña región circular sin vínculo asociado (no tiene el atributo **href**) y el segundo, especifica una región circular más grande con el mismo centro y un vínculo. La combinación de ambos define un anillo cuyo centro es inactivo y cuya corona es activa. El orden de las definiciones de los vínculos es importante, ya que el círculo menor debe prevalecer sobre el círculo mayor.

```

<map name="mapa1">
  <a shape="circle" coords="100,200,50">Vínculo inactivo.</a>
  <a href="anillo.html" shape="circle" coords="100,200,250">Vínculo activo.</a>
</map>

```

Si se utilizara el elemento **AREA**, el atributo **nohref** declararía la región geométrica como una zona sin vínculos asociados.

Mapas de imágenes del lado del servidor.

Los mapas de imágenes en el lado del servidor son útiles en los casos en que el mapa de imágenes es demasiado complicado para un mapa de imágenes en el lado del cliente. Sólo es posible definir un mapa de imágenes del lado del servidor para los elementos:

- **IMG**: debe estar dentro de un elemento **A** con el atributo booleano **ismap**.
- **INPUT**: debe ser del tipo **image**.

Cuando el usuario activa el vínculo haciendo clic sobre la imagen, las coordenadas de pantalla (expresadas como píxeles de pantalla relativos a la imagen) se envían directamente al servidor donde se aloja el documento.

En el siguiente ejemplo, la región activa define un vínculo en el lado del servidor, y un clic en cualquier parte de la imagen hará que las coordenadas del clic sean enviadas al servidor.

```

<p><a href="http://www.dominio.com/cosas/prueba">
  </a></p>

```

Para dar al servidor el lugar del clic, el agente de usuario crea un nuevo URI a partir del URI especificado por el atributo **href** del elemento **A**, añadiendo un '?' seguido de las coordenadas **x** e **y**, separadas por una coma. Una vez hecho esto, se sigue el vínculo especificado por el nuevo URI. En el ejemplo dado, si el usuario hace clic en **x=25, y=27**, el URI creado sería "http://www.dominio.com/cosas/prueba?25,27".

Los agentes de usuario que no ofrezcan al usuario medios para especificar unas coordenadas específicas (Ej. agentes de usuario no gráficos que reciban la entrada por teclado, agentes de usuario por voz, etc.) deberían enviar las coordenadas "0,0" al servidor cuando se activa el vínculo.

10. Hojas de estilo



INTRODUCCIÓN A LAS HOJAS DE ESTILO

En los entornos científicos en que la Web fue concebida, la preocupación estaba más centrada en el contenido de las páginas que en su presentación. Con el tiempo y la expansión de la Web hacia otros ámbitos la importancia de la estética fue aumentando y las personas se vieron forzadas a superar las limitaciones estilísticas del HTML. Si bien la intención era mejorar la presentación de las páginas web, las técnicas creadas para ello han tenido efectos negativos ya que incrementan la complejidad de las páginas web, ofrecen una flexibilidad limitada, sufren de problemas de interoperabilidad, y crean dificultades para las personas con discapacidades.

Entre esas técnicas se incluye:

- La utilización de extensiones propietarias del HTML
- Conversión del texto en imágenes
- Utilización de imágenes para controlar el espacio en blanco
- La utilización de tablas para la organización de las páginas
- Escribir programas en lugar de usar HTML

Las hojas de estilo resuelven estos problemas al mismo tiempo que reemplazan el limitado rango de mecanismos de presentación del HTML. Las hojas de estilo simplifican el código XHTML liberándolo de las responsabilidades de presentación, y dando a autores y usuarios el control sobre la presentación de los documentos, ya que es más fácil especificar la cantidad de espacio entre líneas, el sangrado de las líneas, los colores a utilizar para el texto y el fondo, el tamaño y estilo de las fuentes, y otros muchos detalles.

Por ejemplo, el siguiente estilo declarado en el fichero CSS "estilos.css" hace que en un párrafo el texto sea de color azul ("blue") y rodeado de un borde rojo ("red") continuo ("solid") de 1px de grosor:


```
p.resaltado {
  color : blue;
  border: 1px solid red;
}
```

Esta hoja de estilo puede ser vinculada en el documento HTML a través del elemento **LINK**:

```
<!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="en-US" lang="en-US">
<head>
  <title>Documento HTML de prueba</title>
  <link href="estilos.css" rel="stylesheet" type="text/css" />
</head>

<body>
  <p class="resaltado">El primer párrafo resaltado en azul.</p>
</body>
</html>
```

🔗 [Ver ejemplo de aplicación de estilo \(CD > ejemplos > xhtml > css > ejemplo.html\)](#)

XHTML soporta las siguientes características de hojas de estilo:

- **Colocación flexible de la información de estilo:** Colocar las hojas de estilo en ficheros separados permite una reutilización más fácil. Sin embargo, algunas veces es útil incluir los estilos agrupados en el **HEAD** del documento al que se aplican o en atributos de los elementos a lo largo del cuerpo del documento.
- **Independencia de lenguajes de hojas de estilo específicos:** Se puedan usar desde los lenguajes de hojas de estilo más simples válidos para la mayoría de los usuarios, hasta los más complejos, útiles para una minoría con necesidades muy especializadas.
- **Cascada:** Es la capacidad de algunos lenguajes de hojas de estilo (como CSS) que permiten que las informaciones de estilo provenientes de varias fuentes puedan combinarse. La cascada define una secuencia ordenada de hojas de estilo en la que las reglas de las últimas hojas tienen una prioridad mayor que las de las primeras.
- **Dependencias de los medios:** XHTML permite acceder a las páginas web usando una amplia gama de dispositivos y medios, Ej. pantallas gráficas para ordenadores bajo Windows y Macintosh OS, dispositivos para aparatos de televisión, teléfonos adaptados y dispositivos portátiles PDA, navegadores basados en voz, y dispositivos táctiles Braille.
En un documento XHTML se puede definir a que medios o grupos de medios es aplicable una hoja de estilo, de modo que los agentes de usuario eviten la descarga de hojas de estilo que no sean apropiadas. Una hoja de estilo diseñada para una pantalla puede ser aplicable para una salida impresa, pero es de poca utilidad para los navegadores basados en voz.
- **Estilos alternativos:** Se puede especificar una hoja de estilo preferente y hojas de estilo alternativas que se dirijan a medios o usuarios específicos y que ofrezcan maneras diferentes de ver un documento. Por ejemplo una con fuentes pequeñas y otra con fuentes más grandes para una lectura más cómoda.

Los estilos pueden colocarse:

- **en línea:** dentro de un elemento con el atributo **style**
- **incrustados en cabecera:** dentro del **HEAD** del documento con el elemento **STYLE**
- **en hojas de estilo externas:** vinculadas al documento en el **HEAD** con el elemento **LINK**

Si bien puede usarse cualquier lenguaje de hojas de estilo con XHTML, en este manual utilizaremos el lenguaje de estilo llamado "Hojas de Estilo en Cascada" (Cascading Style Sheets), abreviado CSS.

HOJAS DE ESTILO INCRUSTADAS

Especificación del lenguaje de hojas de estilo por defecto

Los autores deberían especificar el lenguaje de hojas de estilo por defecto de un documento XHTML, a través del elemento **META** dentro de la sección **HEAD**. Para especificar que el valor por defecto es CSS, se debe usar la siguiente declaración:

```
<meta http-equiv="Content-Style-Type" content="text/css">
```

Esto también puede ser establecido con encabezados HTTP:

```
Content-Style-Type: text/css
```

Los agentes de usuario deberían determinar el lenguaje de hojas de estilo por defecto de acuerdo con los siguientes pasos (ordenados de mayor a menor prioridad):

1. La última declaración **META** en el flujo de caracteres que a través de "Content-Style-Type" determina el lenguaje de hojas de estilo por defecto.
2. El último encabezado HTTP en el flujo de caracteres que a través de "Content-Style-Type" determina el lenguaje de hojas de estilo por defecto.
3. Considerar el lenguaje de hojas de estilo por defecto es "text/css".

Utilizar el atributo **style** en un elemento, sin especificar un lenguaje de hojas de estilo por defecto es incorrecto. Las herramientas de creación deberían generar información de lenguaje de hojas de estilo por defecto (normalmente una declaración **META**) de modo que los agentes de usuario no tengan que atenerse al valor por defecto "text/css".

Elemento STYLE

Sintaxis: (etiqueta inicial y final obligatoria)

```
<style>...</style>
```

El elemento **STYLE** es utilizado para crear reglas de hojas de estilo a nivel del documento. Es a nivel del documento, porque se aplicarán a todo el documento HTML, a diferencia del atributo **style** que se comporta como un estilo en línea y sólo afecta a una pequeña parte del documento. Este elemento debe aparecer dentro del elemento **HEAD**, y el código escrito dentro de las etiquetas de apertura y cierre no es HTML, sino CSS (Cascading Style Sheets).

XHTML acepta en la sección **HEAD** de un documento, cualquier número de elementos **STYLE**, los cuáles sirven para declarar reglas de hojas de estilo en la cabecera de dicho documento.

Los agentes de usuario que no soporten hojas de estilo, o que no soporten el lenguaje utilizado por un elemento **STYLE** en particular, deben ocultar los contenidos de dicho elemento y no mostrarlos como parte del texto del documento.

La utilización del elemento **STYLE** permite una gama más amplia de reglas que el uso del atributo **style**. Por ejemplo, con CSS, pueden declararse reglas dentro de un elemento **STYLE** para:

- Todas las apariciones de un elemento XHTML específico (Ej. todos los elementos **P**, **H1**, etc.)
- Todas las apariciones de un elemento XHTML que pertenezcan a una clase específica (cuyo atributo **class** tome cierto valor).
- Las apariciones únicas de un elemento XHTML (cuyo atributo **id** tome cierto valor).

Las reglas de precedencia y herencia de las reglas de estilo dependen del lenguaje de hojas de estilo.

Atributos aceptados		Modelo de contenido	
XHTML Strict		Frameset, Transitional	Contiene a: No acepta:
principales: id, title de lenguaje: lang, xml:lang, dir otros: <u>type</u> * (ContentType), <u>media</u> (MediaDesc), xml:space (= "preserve")		PCDATA	
Desarrollo de atributos			
type <i>type="text/css"</i>	Este atributo especifica el lenguaje de hojas de estilo de los contenidos del elemento y prevalece sobre el lenguaje de hojas de estilo por defecto. El lenguaje de hojas de estilo se especifica como un tipo de contenido (Ej. "text/css"). Los autores deben proporcionar un valor para este atributo ya que no hay un valor por defecto para el mismo.		
media <i>media="screen"</i>	<p>Este atributo especifica el medio destino al que se dirige la información de estilo. Es una lista separada por comas de los descriptores de medios, que permite a los agentes de usuario cargar y aplicar las hojas de estilo de manera selectiva. El valor por defecto de este atributo es "screen" (pantalla), sin embargo, las otras opciones (ver MediaDesc) son: tty, tv, projection, handheld, print, braille, aural, all.</p> <p>Por ejemplo, un elemento H1 se puede representar de manera diferente en una proyección de una reunión (centrado y azul) y cuando se imprime (a la izquierda y negro):</p> <pre><head> <style type="text/css" media="projection"> h1 { text-align:center; color:blue } </style> <style type="text/css" media="print"> h1 { text-align:left; color:black } </style> </head></pre> <p>El control de medios es útil cuando se aplica a hojas de estilo externas, ya que los agentes de usuario pueden obtener de la red únicamente aquellas hojas de estilo que se apliquen el dispositivo actual. Por ejemplo, los navegadores basados en voz pueden evitar la descarga de hojas de estilo diseñadas para la representación visual.</p>		

Ejemplos

Todas las apariciones de un elemento

El siguiente estilo pone un borde alrededor de todos los elementos **H1** y los centrar en la página:

```
<head>
  <style type="text/css">
    h1 {border: 1px solid red; text-align: center}
  </style>
</head>
```

Todas las apariciones de un elemento de una clase específica

El siguiente estilo especifica que el estilo sólo debe aplicarse a los elementos **H1** de la clase "especial":

```
<head>
  <style type="text/css">
    h1.especial {border: 1px solid red; text-align: center}
  </style>
</head>
<body>
  <h1 class="especial">h1 modificado por el estilo "especial".</h1>
  <h1>Encabezado no modificado por el estilo.</h1>
</body>
```

Nota: Al declarar en el elemento **STYLE** la clase, el nombre de la misma va precedido por un punto (.)

Aparición única de un elemento

El siguiente estilo limita su alcance a una sola aparición de **H1**, al cuál se le asignó el atributo **id**:

```
<head>
  <style type="text/css">
    #especial {border: 1px solid red; text-align: center}
  </style>
</head>
<body>
  <h1 id="especial">h1 modificado por el estilo "especial".</h1>
  <h1 class="especial">Encabezado no modificado por el estilo.</h1>
  <h1>Encabezado no modificado por el estilo.</h1>
</body>
```

Nota: Al declarar en el elemento **STYLE** el **id**, el nombre del mismo va precedido por el signo numeral (#)

Utilización de span y estilos

Los elementos **DIV** (en bloque) y **SPAN** (en línea) son útiles ya que no imponen ningún significado presentacional, y combinados con los atributos **id** y **class** permiten extender las posibilidades del XHTML.

En el siguiente ejemplo, utilizamos el elemento **SPAN** para especificar como versalitas ("small-caps") el estilo de fuente de las primeras palabras de un párrafo:

```
<head>
  <style type="text/css">
    span.versales { font-variant: small-caps }
  </style>
</head>
<body>
  <p><span class="versales">Mis palabras</span> en versalitas</p>
</body>
```

Utilización de div y estilos

En otro ejemplo, utilizamos el elemento **DIV** y el atributo **class** para justificar y poner en negrita el texto de una serie de párrafos importantes de un texto. Esta información de estilo puede reutilizarse en otra serie de párrafos importantes repitiendo el atributo **class**:

```
<head>
  <style type="text/css">
    div.importante { text-align: justify; font-weight: 700 }
  </style>
</head>
<body>
  <div class="importante">
    <p>En el año 2001 fue sancionado con tres meses de suspensión y 8000 dólares de multa por la Asociación de Tenistas Profesionales, por dopaje positivo por el uso de metiltestosterona en el torneo de Cincinnati.</p>
  </div>
```

```
<p>El 9 de agosto del 2004 pudo alcanzar la posición número 15 en el ranking mundial. El 19 de enero del 2006 derrotó a Lleyton Hewitt en la segunda ronda del Abierto de Australia.</p>
</div>
</body>
```

Ocultar datos de estilo a los agentes de usuario

Algunos lenguajes de hojas de estilo permiten ocultar el contenido de los elementos **STYLE** a los agentes de usuario no conformes. En CSS el contenido se comenta para asegurarse que los agentes de usuarios antiguos y no conformes no lo representen como texto:

```
<style type="text/css">
<!--
  h1 { color: red }
  p { text-align: justify }
-->
</style>
```

HOJAS DE ESTILO EXTERNAS

La separación de las hojas de estilo con respecto a los documentos XHTML ofrece varias ventajas:

- Los autores y administradores pueden compartir hojas de estilo entre varios documentos y sitios.
- Los autores pueden cambiar la hoja de estilo sin hacer modificaciones en el documento.
- Los agentes de usuario pueden cargar hojas de estilo selectivamente (en función de los medios).

Hojas de estilo preferentes y alternativas

XHTML permite asociar cualquier número de hojas de estilo externas a un documento, y es el lenguaje de hojas de estilo quien define el modo en que interaccionan las mismas (por ejemplo, las reglas de "cascada" de CSS).

Se puede especificar cualquier número de hojas de estilo mutuamente excluyentes llamadas hojas de estilo **alternativas**, que pueden ser seleccionadas por los usuarios según sus preferencias. Por ejemplo, un autor puede especificar una hoja de estilo para pantallas pequeñas y otra para usuarios con poca visión (Ej. con fuentes grandes).

El autor puede especificar que una de las alternativas es una hoja de estilo **preferente** y el agente de usuario debería aplicar la misma a menos que el usuario haya seleccionado una alternativa diferente.

Los autores pueden agrupar varias hojas de estilo alternativas y preferentes bajo un **nombre de estilo común**. Cuando un usuario seleccione un nombre de estilo, el agente de usuario deberá aplicar todas las hojas de estilo con ese nombre, sin poder aplicar una hoja de estilo alternativa con un nombre de estilo diferente.

Los autores también pueden especificar hojas de estilo **persistentes** que los agentes de usuario deben aplicar además de cualquier hoja de estilo alternativa.

Los agentes de usuario deben respetar los descriptores de medios al aplicar cualquier hoja de estilo, y deben también permitir a los usuarios deshabilitar completamente las hojas de estilo del autor, en cuyo caso el agente de usuario no debería aplicar ninguna hoja de estilo persistente ni alternativa.

Especificación de hojas de estilo externas

Como indicamos con anterioridad, las hojas de estilos externas son llamadas a través del elemento **LINK** que ha sido desarrollado en el tema [Vínculos > Elemento LINK](#).

Las hojas de estilo externas se especifican mediante los siguientes atributos del elemento **LINK**:

- **href**: es un URI que indica la localización del archivo de hoja de estilo.
- **type**: indica el lenguaje de la hoja de estilo (text/css). Esto permite al agente de usuario evitar la descarga de una hoja de estilo cuyo lenguaje no soporte.
- **rel**: especifica si la hoja de estilo es persistente, preferente, o alternativa:
- **persistente**: se declara **rel="stylesheet"** sin establecer el atributo **title**.
- **preferente**: se declara **rel="stylesheet"** y se da un nombre a la hoja de estilo mediante el atributo **title**.
- **alternativa**: se declara **rel="alternate stylesheet"** y se da un nombre a la hoja de estilo mediante el atributo **title**.

Los agentes de usuario deberían posibilitar a los usuarios ver la lista de estilos alternativos y escoger uno de ellos en base al nombre del atributo **title**.

Si hay dos o más declaraciones **META** o encabezados HTTP que especifiquen la hoja de estilo preferente, la que prevalece es la última. A estos efectos, se considera que los encabezados HTTP aparecen antes que la sección **HEAD** del documento. Las hojas de estilo preferentes especificadas con un elemento **META** o con encabezados HTTP prevalecen sobre las especificadas con el elemento **LINK**. Si hay dos o más elementos **LINK** que especifiquen una hoja de estilo preferente, el que prevalece es el primero.

Ejemplos

Hoja de estilo persistente

Especificamos una hoja de estilo persistente localizada en el fichero estilos.css:

```
<link href="estilos.css" rel="stylesheet" type="text/css" />
```

Hoja de estilo preferente

Establecemos el atributo **title** y la convertimos en la hoja de estilo preferente del autor:

```
<link href="estilos.css" title="great" rel="stylesheet" type="text/css" />
```

Hoja de estilo alternativa

Añadimos la palabra clave "alternate" al atributo **rel** convirtiéndola en una hoja de estilo alternativa:

```
<link href="estilos.css" title="cool" rel="alternate stylesheet" type="text/css" />
```

Hoja de estilo preferente con el elemento meta

A través del elemento **META** se puede establecer una hoja de estilo como preferente del documento. Si queremos que la hoja preferente sea "great" hay que incluir en el **HEAD** la siguiente línea:

```
<meta http-equiv="Default-Style" content="great" />
```

Hoja de estilo preferente con el encabezado HTTP

La hoja de estilo preferente también se puede especificar mediante encabezados HTTP, lo cuál es equivalente a la declaración en **META**:

```
Default-Style: "great"
```

HOJAS DE ESTILO EN CASCADA

No todos los lenguajes de hojas de estilo soportan la cascada, sin embargo, aquellos que si lo hacen, permiten que se pueda combinar la información de varias fuentes. Para definir una cascada, se especifica una secuencia de elementos **LINK** y/o **STYLE**, y la información se combina en cascada según el orden en que aparecen los elementos en la sección **HEAD**.

En el siguiente ejemplo, especificamos una hoja de estilo persistente, dos hojas alternativas llamadas "great" y una alternativa llamada "letragrande".

```
<link rel="alternate stylesheet" title="great"
      href="peq-base.css" type="text/css" />
<link rel="alternate stylesheet" title="great"
      href="peq-extras.css" type="text/css" />
<link rel="alternate stylesheet" title="letragrande"
      href="tgrandes.css" type="text/css" />
<link rel="stylesheet" href="general.css" type="text/css" />
```

Entonces:

- Si el usuario selecciona el estilo "great", el agente de usuario debe aplicar ambas hojas de estilo externas, junto con la hoja de estilo persistente "general.css".
- Si el usuario selecciona el estilo "tipos grandes", sólo se aplicarán la hoja de estilo alternativa "letragrande.css" y la hoja de estilo persistente "general.css".

Otro ejemplo puede incluir una cascada con elementos **LINK** y **STYLE** juntos:

```
<link rel="stylesheet" href="general.css" type="text/css" />
<link rel="stylesheet" href="estructura.css" type="text/css" />
<style type="text/css">
    p.especial { background: black }
</style>
```

Cascadas dependientes del medio

Una cascada puede incluir hojas de estilo para medios diferentes, ya que **LINK** y **STYLE** pueden usarse con el atributo **media**, en cuyo caso el agente de usuario deberá filtrar aquellas hojas de estilo que no se apliquen al medio actual.

En el siguiente ejemplo, definimos una cascada en la cual se incluyen versiones de la hoja de estilo "general" para impresión, pantalla y navegadores basados en voz. La hoja de estilo "resultados" se aplica a todos los medios y el fondo definido por el elemento **STYLE** se usa para impresoras y pantallas, pero no para la representación auditiva.

```
<link rel="stylesheet" media="aural" href="general-aural.css" type="text/css" />
<link rel="stylesheet" media="screen" href="general-screen.css" type="text/css"/>
<link rel="stylesheet" media="print" href="general-print.css" type="text/css" />
<link rel="stylesheet" href="resultados.css" type="text/css" />
<style media="screen, print" type="text/css">
    p.especial { background: black }
</style>
```

11. Fuentes y separadores horizontales



Hay ciertos atributos que han sido desaprobados para dar formato visual a los elementos, ya que la misma tarea puede realizarse con las hojas de estilo. Algunos de estos atributos son: **background** (color de fondo), **align** (alineación), **clear** (control del flujo), **strike** y **s** (tachado), **u** (subrayado), **font** (cambio de tamaño y fuente con los atributos **size**, **face** y **color**), **basefont** (tamaño base de fuente).

ELEMENTOS TT, I, B, BIG, SMALL

Sintaxis: (etiqueta inicial y final obligatoria)

```
<tt>...</tt>
<i>...</i>
<b>...</b>
<big>...</big>
<small>...</small>
```

La representación de estos elementos de estilo de fuente depende del agente de usuario, pero de forma informativa, son representados como:

- **TT**: el texto se muestra en una fuente de teletipo o ancho fijo.
- **I**: el texto se muestra en estilo de texto itálica.
- **B**: el texto se muestra en estilo de texto negrita (puede reemplazarse con **font-weight** en CSS).
- **BIG**: el texto se muestra en una fuente "grande". El cambio lo determina el navegador, y si la fuente ya está en el tamaño máximo, este elemento no tendrá ningún efecto.
- **SMALL**: el texto se muestra en una fuente "pequeña". El cambio lo determina el navegador, y si la fuente ya está en el tamaño mínimo, este elemento no tendrá ningún efecto.

Generalmente los elementos **TT**, **CODE**, **KBD** y **SAMP** son mostrados con la misma fuente en el mismo navegador.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Inline (acepta elementos en línea)	

Ejemplo

El siguiente código abarca los formatos descriptos:

```
<b>negrita</b>, <i>itálica</i>, <b><i>negrita itálica</i></b>, <tt>texto de teletipo</tt>, texto normal, <big>texto grande</big> y <small>texto pequeño</small>.
```

Lo que sería representado de la siguiente forma:

negrita, *itálica*, ***negrita itálica***, texto de teletipo, y texto grande y pequeño.

Nota: Siempre es posible lograr una variedad mayor de efectos de fuentes usando hojas de estilo.

ELEMENTO HR

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<hr />
```

El elemento **HR** se utiliza para representar una regla o línea horizontal. La cantidad de espacio vertical insertado entre el separador y el contenido que le rodea depende del agente de usuario. El flujo de texto e imágenes se detiene en la línea actual, la regla se muestra en la línea siguiente, y luego el flujo de texto a imágenes continúa en la próxima línea. La apariencia de la línea puede ser cambiada con hojas de estilo.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Empty	

12. Marcos



INTRODUCCIÓN A LOS MARCOS

Los **marcos** o **frames** permiten presentar documentos con vistas múltiples, que pueden ser ventanas o subventanas independientes. Las vistas múltiples ofrecen una manera de mantener cierta información visible mientras otras vistas se desplazan o se sustituyen. Por ejemplo, dentro de una misma ventana:

- un marco podría mostrar un gráfico estático, como el logo de la web.
- un segundo marco un menú de navegación
- y un tercero el documento principal que puede ser desplazado, o reemplazado al navegar por el segundo marco.

Aquí tenemos un documento simple con marcos:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
  <title>Documento de prueba con marcos.</title>
</head>

<frameset cols="20%, 80%">
  <frameset rows="100, 200">
    <frame src="marco1.gif" />
    <frame src="marco2.html" />
  </frameset>
  <frame src="marco3.html" />

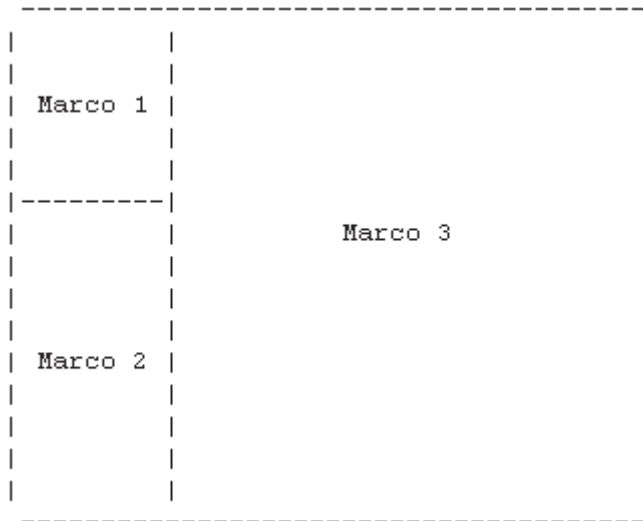
  <noframes>
    <p>Este conjunto de marcos contiene:</p>
    <ul>
      <li>
```

```

    <li><a href="marco2.html">Menú del sitio</a>
    <li><a href="marco3.html">Contenido del sitio</a>
  </ul>
</noframes>
</frameset>
</html>

```

Esto podría crear una disposición de marcos como la siguiente:



Si el agente de usuario no puede mostrar marcos o está configurado para no mostrarlos, representará los contenidos del elemento **NOFRAMES**.

DISPOSICIÓN DE LOS MARCOS

Los documentos con marcos tienen una estructura diferente a los documentos sin marcos. Ambos poseen una sección **HEAD**, sin embargo, los documentos con marcos reemplazan la sección **BODY** por **FRAMESET**.

La sección **FRAMESET** de un documento especifica la disposición de las vistas en la ventana principal del agente de usuario. Además, la sección **FRAMESET** puede contener un elemento **NOFRAMES** que proporcione contenido alternativo para los agentes de usuario que no soporten marcos o que estén configurados para no mostrar marcos.

Los elementos que normalmente podrían colocarse en el elemento **BODY** no deben aparecer antes del primer elemento **FRAMESET**, o el **FRAMESET** no será tenido en cuenta.

Elemento FRAMESET

Sintaxis: (etiqueta inicial y final obligatoria)

```
<frameset>...</frameset>
```

El elemento **FRAMESET** sirve como contenedor de un grupo de marcos. Especifica la organización de la ventana principal del usuario en términos de subespacios rectangulares. Este elemento determina el número de marcos, si están configurados como columnas o filas, y establece el espaciado en cada caso. La configuración en filas o columnas se establece a través de los atributos **rows** y **cols** (no se puede usar ambos en forma simultánea).

Este elemento se usa en reemplazo del elemento **BODY**, el cuál no se puede utilizar bajo ningún concepto y debe contener otros elementos **FRAMESET**, **FRAME** y el elemento **NOFRAMES**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title eventos de ventana: onload (Script), onunload (Script) otros: <u>cols</u> (MultiLength), <u>rows</u> (MultiLength),		(FRAMESET FRAME NOFRAMES)*	
Desarrollo de atributos			
rows <i>rows="5, *, 20%"</i>	Este atributo especifica el número de marcos horizontales. Es una lista de longitudes separadas por comas y encerrado entre comillas que indica el alto de cada fila. El número de valores en la lista determina el número de marcos. El valor por defecto es 100%, que significa una fila. Si se declaran 3 valores, entonces se esperan 3 filas. Los valores pueden ser: <ul style="list-style-type: none"> • un porcentaje del ancho de la ventana • un número entero en píxeles • una longitud relativa indicada con un asterisco (*). El asterisco se usa para asignar espacio de manera proporcional. Por ejemplo: <code><frameset cols="*, *, 50%"></code> creará tres filas, una que ocupe el 50% y dos de 25%. El valor 3* tiene tres veces la proporción de un solo *. 		
cols <i>cols="5, *, 20%"</i>	Este atributo especifica el número de marcos verticales. Es una lista de longitudes separadas por comas y encerrado entre comillas que indica el ancho de cada columna. El número de valores en la lista determina el número de marcos. El valor por defecto es 100%, que significa una columna. Si se declaran 3 valores, entonces se esperan 3 marcos. Los valores pueden ser: <ul style="list-style-type: none"> • un porcentaje del ancho de la ventana • un número entero en píxeles • una longitud relativa indicada con un asterisco (*). El asterisco se usa para asignar espacio de manera proporcional. Por ejemplo: <code><frameset cols="*, *, 50%"></code> creará tres columnas, una que ocupe el 50% y dos de 25%. El valor 3* tiene tres veces la proporción de un solo *. 		

Filas y columnas

Cuando se establece el atributo **rows** (filas) se define el número de subespacios horizontales. Cuando se establece el atributo **cols** (columnas) se define el número de subespacios verticales. Si no se establece:

- el atributo **rows**, cada columna se extiende a lo largo de toda la longitud de la página.
- el atributo **cols**, cada fila se extiende a lo largo de toda la ancho de la página.
- ninguno de los dos atributos, el marco tiene exactamente el mismo tamaño que la página.

Los marcos se crean de izquierda a derecha para las columnas y de arriba a abajo para las filas. Cuando se especifican ambos atributos, las vistas se crean de izquierda a derecha en la fila superior, de izquierda a derecha en la segunda fila, etc.

Ejemplos

Ejemplo 1: Dividimos la pantalla verticalmente en dos partes iguales creando una parte superior e inferior:

```
<frameset rows="50%, 50%">
  ... contenidos del frameset ...
</frameset>
```

Ejemplo 2: Creamos tres columnas: la primera tiene un ancho fijo de 210 píxeles (útil para incluir una imagen de tamaño conocido), la segunda recibe el 20% del espacio restante, y la tercera el 80% del espacio restante:

```
<frameset cols="1*,210,4*">
... contenidos del frameset ...
</frameset>
```

Ejemplo 3: Dividimos el espacio en 4 filas y suponiendo que la ventana del navegador tiene una altura de 1000 píxeles:

- Primera fila: se asigna el 30% de la altura total (300 píxeles).
- Segunda fila: se especifica una altura exacta de 400 píxeles. Esto deja 300 píxeles para repartir entre los otros dos marcos.
- Tercer fila: se especifica la altura "*" (equivalente a 1*), lo que le da una altura de 100px.
- Cuarta fila: se ha especificado como "2*", de modo que es el doble de alto que el tercer marco, dejándolo en 200 píxeles.

```
<frameset rows="30%,400,*,2*">
... contenidos del frameset ...
</frameset>
```

Las longitudes absolutas que no sumen el 100% del espacio real disponible deberían ser ajustadas por los agentes de usuario. Cuando sobre espacio, el espacio sobrante debería repartirse proporcionalmente entre cada vista. Cuando falte espacio, debería reducirse cada vista en función de la relación entre el espacio especificado y el espacio total.

Anidamiento de grupos de marcos

Los grupos de marcos pueden anidarse hasta cualquier nivel.

Ejemplo

El **FRAMSET** exterior divide el espacio disponible en tres columnas iguales, y luego un FRAMESET interior divide la segunda columna en dos filas de alturas diferentes:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
  <title>Documento de prueba con marcos.</title>
</head>

<frameset cols="30%,30%,40%">
  ... contenidos de la primer columna ...
  <frameset rows="50%,50%">
    ... contenidos de la segunda columna, primer fila ...
    ... contenidos de la segunda columna, segunda fila ...
  </frameset>
  ... contenidos de la tercer columna ...
</frameset>
</html>
```

Compartir datos entre marcos

Los autores pueden compartir datos entre varios marcos incluyendo estos datos a través de un elemento **OBJECT**. Los autores deberían incluir el elemento **OBJECT** en el elemento **HEAD** del documento con marcos y darle un nombre con el atributo **id**. Cualquier documento que sea el contenido de un marco del documento con marcos puede hacer referencia a este identificador.

Ejemplo

El siguiente ejemplo ilustra cómo podría hacer referencia un script a un elemento **OBJECT** definido para todo

un grupo de marcos:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
  <title>Documento de prueba con marcos y un object.</title>
  <!-- ;Este OBJECT no se representa! -->
  <object id="datos" data="datos.dat"></object>
</head>
<frameset>
  <frame src="interior.html" name="interior" />
</frameset>
</html>
```

El documento interior.html tendría el siguiente código:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
  <head>
    <title>Página interior</title>
  </head>
  <body>
    ... texto del documento...
    <script type="text/javascript">parent.datos.mipropiedad</script>
    ... resto del documento...
  </body>
</html>
```

Elemento FRAME

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<frame />
```

El elemento **FRAME** es utilizado para crear un marco, que es una ventana dentro de otra ventana. Define los contenidos y la apariencia de un marco dado. Este elemento sólo puede aparecer dentro del elemento **FRAMESET**.

Un elemento **frame muestra contenido**, incluyendo formularios, imágenes, tablas, etc. La única forma de asignar contenido a un marco es asignar una URL a través del atributo **src**. Aquel contenido que pueda verse en la URL referenciada, también será visualizado dentro del **FRAME**.

Tal como indicamos con anterioridad, los marcos pueden tener una configuración en filas (horizontal) o en columnas (vertical), pero no ambas a la vez. Esta configuración se establece con los atributos **rows** y **cols** respectivamente del elemento **FRAMESET**. El orden de los elementos **FRAME** indica el orden de los marcos. Si la configuración es horizontal y con dos filas, el primer marco será el de arriba y el segundo el de abajo. Si la configuración es vertical, el orden es de izquierda a derecha.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title otros: <code>frameborder</code> ("1" "0"), <code>longdesc</code> (URI), <code>marginheight</code> (Pixels), <code>marginwidth</code> (Pixels), <code>noresize</code> ("noresize"), <code>scrolling</code> ("yes" "no" "auto"), <code>src</code> (URI)		Empty	
Desarrollo de atributos			

Atributos aceptados	Modelo de contenido
name <i>name="unnombre"</i>	Este atributo asigna un nombre al marco actual, el cuál puede utilizarse como destino de vínculos subsiguientes.
longdesc <i>longdesc="URI"</i>	Este atributo especifica un vínculo a una descripción larga del marco, descripción que debería complementar la descripción corta proporcionada por el atributo title , y puede ser particularmente útil para agentes de usuario no visuales.
src <i>src="URI"</i>	Este atributo especifica la localización de los contenidos iniciales que contendrá el marco.
noresize <i>noresize="noresize"</i>	Este atributo booleano le dice al agente de usuario que la ventana del marco no debe ser redimensionable.
scrolling <i>scrolling="auto"</i> <i>scrolling="yes"</i> <i>scrolling="no"</i>	Este atributo especifica si deben aparecer o no barras de desplazamiento horizontal o vertical en el marco. Si el contenido es más largo que el marco, entonces permitirá desplazarse hacia arriba y abajo, o hacia la derecha e izquierda para ver todos los contenidos. Los valores posibles son: <ul style="list-style-type: none"> auto: Este valor le dice al agente de usuario que proporcione mecanismos de desplazamiento en la ventana del marco cuando sea necesario. Este es el valor por defecto. yes: Este valor le dice al agente de usuario que siempre proporcione mecanismos de desplazamiento en la ventana del marco. no: Este valor le dice al agente de usuario que nunca proporcione mecanismos de desplazamiento en la ventana del marco.
frameborder <i>frameborder="1"</i> <i>frameborder="0"</i>	Este atributo proporciona información al agente de usuario sobre el borde del marco, sus valores son: <ul style="list-style-type: none"> 1: Este valor le dice al agente de usuario que dibuje un separador entre este marco y todos los marcos adyacentes. Este es el valor por defecto. 0: Este valor le dice al agente de usuario que no dibuje un separador entre este marco y todos los marcos adyacentes. Obsérvese que aún se puede dibujar un separador junto a este marco si así se especifica para otros marcos.
marginwidth <i>marginwidth="10px"</i>	Este atributo especifica la cantidad de espacio que debe dejarse entre los contenidos del marco en sus márgenes izquierdo y derecho. El valor debe ser mayor o igual que cero (píxeles).
marginheight <i>marginheight="2px"</i>	Este atributo especifica la cantidad de espacio que debe dejarse entre los contenidos del marco en sus márgenes superior e inferior. El valor debe ser mayor o igual que cero (píxeles).

Especificación de los contenidos iniciales de un marco

El atributo **src** establece el documento inicial que contendrá el marco.

Viendo el siguiente ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
  <title>Documento de prueba con marcos.</title>
</head>

<frameset cols="20%, 80%">
```

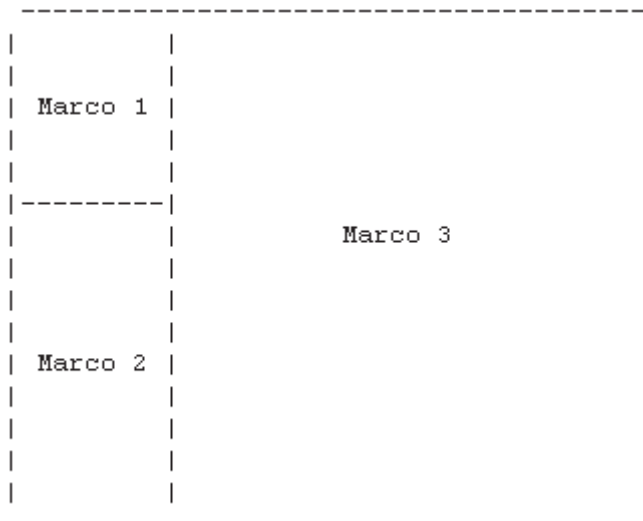
```

<frameset rows="100, 200">
  <frame src="marco1.gif" />
  <frame src="marco2.html" />
</frameset>
<frame src="marco3.html" />

<noframes>
  <p>Este conjunto de marcos contiene:</p>
  <ul>
    <li>
    <li><a href="marco2.html">Menú del sitio</a>
    <li><a href="marco3.html">Contenido del sitio</a>
  </ul>
</noframes>
</frameset>
</html>

```

Esto podría crear una disposición de marcos como la siguiente, donde el agente de usuario debe cargar cada archivo en una vista separada. Los contenidos de un marco no deben estar en el mismo documento que la definición del marco.



Representación visual de un marco

El siguiente ejemplo ilustra el uso de los atributos decorativos de **frame**:

- El marco 1 dejará espacio en blanco alrededor de sus contenidos (inicialmente, un fichero de imagen) y no permitirá barras de desplazamiento.
- El marco 2 no será redimensionable.
- El marco 3 no dibujará los bordes (por defecto).


```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
    <title>Documento de prueba con marcos.</title>
</head>

<frameset cols="20%, 80%">
    <frameset rows="100, 200">
        <frame src="marco1.gif" marginwidth="10" marginheight="15" scrolling="no" />
        <frame src="marco2.html" noresize="noresize" />
    </frameset>
    <frame src="marco3.html" frameborder="0"/>
</frameset>
</html>
```

Descripciones largas de marcos

El atributo **longdesc** permite hacer los documentos con marcos más accesibles a las personas que utilizan agentes de usuario no visuales, ya que designa un recurso que proporciona una descripción larga del marco. Esta descripción larga asociada con los marcos se refiere al marco, y no a los contenidos del mismo. Como los contenidos pueden variar con el tiempo, la descripción larga inicial podría ser inapropiada para los contenidos posteriores del marco. En particular, los autores no deberían incluir una imagen como único contenido de un marco.

El siguiente documento con marcos describe dos marcos. El marco izquierdo contiene un menú y el marco derecho contiene inicialmente la imagen de un paisaje campestre:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
    <title>Documento de prueba con marcos incorrectos.</title>
</head>

<frameset cols="20%, 80%">
    <frame src="contenido_fijo.html" />
    <frame src="paisaje_campestre.gif" longdesc="desc_paisaje.html" />
</frameset>
</html>
```

Obsérvese que la imagen ha sido incluida en el marco independientemente de cualquier elemento HTML, de modo que el autor no tiene ninguna manera de especificar un texto alternativo aparte de usar el atributo **longdesc**. Si los contenidos del marco derecho cambian (Ej. el usuario selecciona el paisaje montaños), los usuarios no tendrán acceso textual a los nuevos contenidos del marco.

Por tanto, los autores no deberían poner una imagen en un marco directamente. En su lugar, la imagen debería especificarse en un documento HTML independiente, en el cual se podría adjuntar el texto alternativo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
    <title>Documento de prueba con marcos incorrectos.</title>
</head>

<frameset cols="20%, 80%">
    <frame src="contenido_fijo.html" />
    <frame src="paisaje_campestre.html"/>
</frameset>
</html>
```

Luego, en el archivo `paisaje_campestre.html` se coloca la imagen:

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

<html>
<head>
  <title>Paisaje campestre argentino.</title>
</head>

<body>
  ... texto del documento...
  <object data="paisaje_campestre.gif" type="image/gif">Hermoso paisaje campestre
  en la pampa argentina.</object>
</body>
</html>
```

INFORMACIÓN SOBRE EL MARCO DESTINO

Atributo target

target

Este atributo especifica el nombre de un marco en el que debe abrirse un documento. Al asignar un nombre a un marco por medio del atributo **name**, los autores pueden referirse a él como el "destino" (**target**) de los vínculos definidos por otros elementos como **A**, **LINK**, **AREA** y **FORM**. **Este atributo no está permitido en XHTML Strict.**

Ejemplo

Veamos el siguiente ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
  <title>Documento de prueba con marcos.</title>
</head>

<frameset cols="20%, 80%">
  <frame src="contenido_fijo.html" name="fijo" />
  <frame src="contenido_dinamico.html" name="dinamico" />
</frameset>
</html>
```

Luego, en el archivo contenido_dinamico.html se hace un vínculo al marco llamado "dinamico":

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

<body>
  ... texto del documento...
  <p>Ir a la <href="galeria1.html" target="dinamico">Galería 1</a>
  ... texto del documento...
  <p>Ir a la <href="galeria2.html" target="dinamico">Galería 2</a>
  ... resto del documento...
</body>
</html>
```

Si se activa cualquiera de los dos vínculos se abre un nuevo documento en el marco llamado "dinamico",

mientras que el otro marco, "fijo", mantiene sus contenidos iniciales.

Nombres de marcos destino

%FrameTarget; indica los nombres de marco destino que, exceptuando los nombres reservados enumerados a continuación, deben empezar con un carácter alfabético (a-zA-Z):

- **_blank**: El documento designado debería cargarse en una ventana nueva y sin nombre.
- **_self**: El documento designado debería cargarse en el mismo marco que el elemento que hace referencia a este destino.
- **_parent**: El documento designado debería ser cargado en el **FRAMESET** padre inmediato del marco actual. Este valor es equivalente a **_self** si el marco actual no tiene padre.
- **_top**: El documento designado debería cargarse en la ventana original completa (cancelando así todos los demás marcos). Este valor es equivalente a **_self** si el marco actual no tiene padre.

Destino de los vínculos por defecto

Cuando muchos vínculos en un mismo documento designan al mismo destino, se puede especificar el destino una sola vez para que no sea necesario incluir el atributo **target** en todos los elementos. Esto se hace estableciendo el atributo **target** del elemento **BASE**.

El ejemplo anterior con el elemento base quedaría como sigue:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Página interior dinámica</title>
  <base href="http://www.dominio.com/galerias/" target="dinamico" />
</head>

<body>
  ... texto del documento...
  <p>Ir a la <href="galeria1.html">Galería 1</a>
  ... texto del documento...
  <p>Ir a la <href="galeria2.html">Galería 2</a>
  ... resto del documento...
</body>
</html>
```

Semántica de marcos destino

Los agentes de usuario deberían determinar el marco destino en el que cargar un recurso de acuerdo con las siguientes reglas de precedencia (ordenadas de mayor a menor prioridad):

1. Si un elemento especifica en su atributo **target** un marco conocido, cuando se activa el vínculo el recurso designado por el elemento debería cargarse en el marco destino.
2. Si un elemento no tiene el atributo **target** establecido pero el elemento **BASE** sí lo tiene, el atributo **target** del elemento **BASE** determina el marco.
3. Si ninguno hace referencia a un destino, el recurso designado por el elemento debería cargarse en el marco que contiene al elemento.
4. Si alguno de los atributos **target** se refiere a un marco desconocido F, el agente de usuario debería crear una ventana y marco nuevos, asignar el nombre F al marco, y cargar el recurso designado por el elemento en el nuevo marco.

Al no haber garantía de que un nombre de destino de un marco sea único, se debe seguir la siguiente práctica para encontrar un marco con un nombre de destino dado:

1. Si el nombre de destino es una palabra reservada de las descritas en el texto normativo, se aplica según se describe.
2. En caso contrario, se realiza una búsqueda en la jerarquía de marcos de la ventana que contenía el vínculo. Se usa el primer marco cuyo nombre concuerde exactamente.
3. Si no se encuentra ningún marco en (2), se aplica el paso 2 a todas las ventanas, desde el primer plano hasta el fondo. Se detiene la búsqueda en cuanto se encuentre un marco con exactamente el mismo nombre.
4. Si no se encuentra ningún marco en (3), se crea una ventana nueva y se le asigna el nombre de destino.

CONTENIDO ALTERNATIVO

Los autores deberían proporcionar contenido alternativo para aquellos agentes de usuario que no soporten marcos o que estén configurados para no mostrar marcos.

Elemento noframes

Sintaxis: (etiqueta inicial y final obligatoria)

```
<noframes>...</noframes>
```

El elemento **NOFRAMES** se utiliza para mostrar un mensaje alternativo en aquellos navegadores que no reconozcan marcos (siempre) o que estén configurados para no mostrarlos. Usualmente este mensaje advierte al usuario que se requieren marcos y le da una dirección para acceder al contenido. Si el navegador reconoce marcos, el contenido de **NOFRAMES** no es mostrado. Este elemento debe ser colocado inmediatamente después del primer uso del elemento **FRAMESET**.

El elemento **NOFRAMES** es parte del DTD transicional y DTD de documentos con marcos. En un documento que use el DTD de documentos con marcos, **NOFRAMES** se puede usar al final de la sección **FRAMESET** del documento.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transicional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout		Flow (acepta elementos en línea y en bloque)	Otros NOFRAMES

Ejemplo

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
<html>
<head>
  <title>Documento de prueba con marcos.</title>
</head>

<frameset cols="20%, 80%">
  <frameset rows="100, 200">
    <frame src="marco1.gif" />
    <frame src="marco2.html" />
  </frameset>
  <frame src="marco3.html" />
</frameset>

<noframes>
  <p>Este conjunto de marcos contiene:</p>
  <ul>
```

```

<li>
<li><a href="marco2.html">Menú del sitio</a>
<li><a href="marco3.html">Contenido del sitio</a>
</li>
</ul>
</noframes>
</frameset>
</html>

```

NOFRAMES se puede usar en un documento que es el origen de un marco y que usa el DTD transicional, para explicar el propósito del documento en los casos en que éste se vea fuera del grupo de marcos o con un agente de usuario que no soporte marcos.

MARCOS EN LÍNEA

Elemento IFRAME

Sintaxis: (etiqueta inicial y final obligatoria)

```
<iframe>...</iframe>
```

El elemento **IFRAME** permite insertar un marco en línea dentro de un bloque de texto, lo cuál es muy similar a insertar un objeto mediante un elemento **OBJECT**: ambos permiten insertar un documento HTML en medio de otro, ambos pueden alinearse con el texto circundante, etc.

A diferencia de los elementos **FRAMESET**, **FRAME** y **NOFRAMES**, el elemento **IFRAME** sólo puede ser colocado dentro del elemento **BODY** y nunca dentro de **FRAMESET**.

Un elemento **IFRAME** muestra contenido, incluyendo formularios, imágenes, tablas, etc. La única forma de asignar contenido a un **IFRAME** es asignar una URL a través del atributo **src**. Aquel contenido que pueda verse en el URI referenciado, también será visualizado dentro del **FRAME**.

Se puede colocar un mensaje entre las etiquetas de apertura y cierre del elemento, que sólo será mostrado por aquellos navegadores que no reconozcan los marcos en línea.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
	principales: id, class, style, title otros: <u>longdesc</u> (URI), src (URI), frameborder ("1" "0"), marginwidth (Pixels), marginheight (Pixels), scrolling ("yes" "no" "auto"), <u>height</u> (Lenght), <u>width</u> (Lenght), align (ImgAlign), <u>name</u> (NMTOKEN)	Flow (acepta elementos en línea y en bloque)	
Desarrollo de atributos			
longdesc <u>longdesc</u> ="URI"	Este atributo especifica un vínculo a una descripción larga del marco. Esta descripción debería servir como complemento de la descripción corta que proporciona el atributo title , y es útil para los agentes de usuario no visuales.		

name <i>name="unnombre"</i>	Este atributo asigna un nombre al marco actual, el cuál puede utilizarse como el destino de vínculos subsiguientes.
width <i>width="200px"</i>	El ancho del marco en línea.
height <i>height="200px"</i>	El alto del marco en línea.

Ejemplo

Para aquellos agentes de usuario que soporten marcos, el siguiente ejemplo colocará un marco en línea rodeado por un borde en medio del texto.

```
<iframe src="archivo.html" width="400" height="500" scrolling="auto"
frameborder="1">
  [Su agente de usuario no soporta marcos o está configurado para no mostrarlos.
Sin embargo, puede visitar <a href="archivo.html">el documento relacionado.</a>]
</iframe>
```

Los marcos en línea no pueden ser redimensionados (y por lo tanto no tienen un atributo **noresize**).

13. Formularios



INTRODUCCIÓN A LOS FORMULARIOS

Un formulario XHTML es una sección de un documento que contiene elementos especiales llamados **controles** (casillas de verificación, radiobotones, menús, etc.) y **rótulos** (labels) en esos controles. Cuando los usuario completan un formulario, modifican sus controles para luego enviar los datos entrados en el formulario a un agente para que lo procese (Ej. a un servidor web, a un servidor de correo, etc.)

A continuación mostramos un formulario simple con rótulos, radiobotones y botones para limpiar el formulario o enviarlo:

```
<form action="http://www.dominio.com/contratar" method="post">
  <label for="nombre">Nombre: </label>
    <input type="text" id="nombre" /><br />
  <label for="apellido">Apellido: </label>
    <input type="text" id="apellido" /><br />
  <label for="email">email: </label>
    <input type="text" id="email" /><br />
  <input type="radio" name="sexo" value="Varón" /> Varón<br />
  <input type="radio" name="sexo" value="Mujer" /> Mujer<br />
  <input type="submit" value="Enviar" /> <input type="reset" />
</form>
```

CONTROLES

Los usuarios interaccionan con los formularios a través de los **controles**.

El "**nombre de control**" de un control está dado por su atributo **name**, y el "campo de acción" o alcance del atributo **name** de un control contenido en un elemento **FORM** es el elemento **FORM**.

Cada control tiene un valor inicial y un valor actual, que son cadenas de caracteres.

- En general, el "**valor inicial**" de un control puede especificarse con el atributo **value** del elemento de control. Sin embargo, en un elemento **TEXTAREA** viene dado por sus contenidos, y en un elemento **OBJECT** de un formulario está determinado por la implementación del objeto.
- El "**valor actual**" del control en primer lugar es igual al valor inicial, pero luego puede ser modificado a través de la interacción con el usuario y mediante **SCRIPTS**.

El valor inicial de un control no cambia, entonces si se resetea un formulario, el valor actual de cada control es reemplazado por su valor inicial. Si el control no tiene un valor inicial, el efecto de una reinicialización del formulario sobre ese control es indefinido.

Cuando se envía un formulario para su procesamiento, para algunos controles se empareja su nombre con su valor actual, y estas parejas se envían con el formulario. Aquellos controles cuyas parejas nombre/valor se envían se llaman controles con éxito.

Tipos de controles

XHTML define los siguientes tipos de controles:

botones

Se pueden crear tres tipos de botones:

- **botones de envío (submit buttons):** Cuando es activado envía un formulario. Un formulario puede contener más de un botón de envío.
- **botones de reinicialización (reset buttons):** Cuando se activa se resetean todos los controles a sus valores iniciales.
- **botones pulsadores (push buttons):** No tienen un comportamiento por defecto, cada uno puede tener asociados scripts en el lado del cliente a través del atributo **event** del elemento. Cuando ocurre un evento (Ej. el usuario aprieta el botón, lo suelta, etc.), se acciona el script asociado. Los autores deberían especificar el lenguaje de programación del script de un botón pulsador a través de una declaración de scripts por defecto (con el elemento **META**).

Nota: Se pueden crear botones con el elemento **BUTTON** o el elemento **INPUT**. Sin embargo **BUTTON** ofrece posibilidades más ricas de representación que **INPUT**.

casillas de verificación (checkboxes)

Las casillas de verificación (y los radiobotones) son interruptores de encendido/apagado que pueden ser conmutados por el usuario. Una casilla de verificación está "marcada" cuando se establece el atributo **checked** del elemento de control. Cuando se envía un formulario, solamente pueden tener éxito los controles de casillas de verificación que estén marcadas.

Varias casillas de verificación de un formulario pueden compartir el mismo nombre de control. Así, las casillas de verificación permiten a los usuarios elegir varios valores para la misma propiedad. Para crear un control de casilla de verificación se utiliza el elemento **INPUT**.

radiobotones (radio buttons)

Los radiobotones son como las casillas de verificación, excepto que cuando varios comparten el mismo nombre de control, son mutuamente exclusivos, es decir que cuando uno está "encendido", todos los demás con el mismo nombre se "apagan". Para crear un control de radiobotón se usa el elemento **INPUT**.

Siempre uno de los radiobotones está marcado, ya que si ninguno de los elementos **<input>** de especifica '**checked**', el agente de usuario debe marcar el primer radiobotón del conjunto.

menús (menus)

Los menús ofrecen opciones entre las cuales elegir, se crean con el elemento **SELECT** en combinación con los elementos **OPTGROUP** y **OPTION**.

entrada de texto (text input)

Los autores pueden crear dos tipos de controles que permiten a los usuarios introducir textos, los cuáles pueden ser de una sola línea (**INPUT**) o de varias líneas (**TEXTAREA**). En ambos casos, el texto introducido se convierte en el valor actual del control.

selección de ficheros (file select)

Este tipo de control realizado con **INPUT** permite al usuario elegir ficheros de modo que sus contenidos puedan ser enviados con un formulario.

controles ocultos (hidden controls)

Se pueden crear controles con el elemento **INPUT** que no se muestran pero cuyos valores se envían con un formulario. Sirven para almacenar información entre intercambios cliente/servidor que de otro modo se perdería debido a la naturaleza no persistente del protocolo HTTP.

controles tipo objeto (object controls)

Se pueden insertar objetos genéricos con el elemento **OBJECT** en los formularios de modo que los valores asociados se envíen junto con los demás controles.

Los elementos utilizados para crear controles aparecen normalmente dentro de un elemento **FORM**, pero también pueden aparecer fuera de la declaración de un elemento **FORM** cuando se utilizan para construir interfaces de usuario (son eventos intrínsecos que no pueden ser controles con éxito).

ELEMENTO FORM

Sintaxis: (etiqueta inicial y final obligatoria)

```
<form>...</form>
```

El elemento **FORM** es utilizado para delimitar el inicio y final de un formulario y sirve como contenedor de controles (campos). **Control** es un término técnico que hace referencia a los elementos que pueden utilizarse dentro de un formulario para obtener información. Dicha información obtenida hace referencia a los contenidos o parámetros del formulario y son una colección de parejas nombre/valor.

Los cuatro elementos que pueden utilizarse para construir un formulario son:

- **INPUT**
- **BUTTON**
- **SELECT**
- **TEXTAREA**

Un formulario puede contener texto y código (párrafos, listas, etc.) además de controles de formulario.

El concepto es que el usuario completa las secciones del formulario como respuesta a un pedido de información, y hace clic en el botón de envío. Frente a esto, los contenidos del formulario son enviados para procesar, hacia otra página en el mismo sitio web. Sin embargo, se pueden enviar a la misma página, o a otra ventana o marco.

Existen dos atributos requeridos en el elemento **FORM**:

1. **action**: dice a donde serán enviados los datos del formulario.
2. **method**: especifica como realizar el envío.

Se recomienda utilizar un solo formulario por página, y dividir los formularios largos entre dos o más páginas.

El elemento **FORM** actúa como contenedor de controles y especifica:

- La disposición ("layout") del formulario (dada por los contenidos del elemento).
- El programa que manejará el formulario completado y enviado (el atributo **action**). El programa receptor debe ser capaz de interpretar las parejas nombre/valor para poder hacer uso de ellas.
- El método por el cual se enviarán los datos del usuario al servidor (el atributo **method**).
- Una codificación de caracteres que debe ser aceptada por el servidor para poder manejar este formulario (el atributo **accept-charset**).

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir de teclado: accesskey, tabindex eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario: eventos de formularios: onsubmit, onreset otros: <u>action</u> * (URI), <u>method</u> ("get" "post"), <u>enctype</u> (ContentType = "application/x-www-form-urlencoded"), <u>accept</u> (ContentTypes), <u>accept-charset</u> (Charsets)	name (NMTOKEN), target (FrameTarget)	Block (acepta elementos en bloque y scripts)	FORM
Desarrollo de atributos			
action <i>action="URI"</i>	Este atributo obligatorio es un URI que especifica un agente procesador de formularios. El comportamiento del agente de usuario frente a un valor diferente de un URI HTTP es indefinido.		
method <i>method="get"</i> <i>method="post"</i>	Este atributo especifica qué método HTTP se usará para enviar el conjunto de datos del formulario al agente procesador. Los valores posibles son: <ul style="list-style-type: none"> • "get" (valor por defecto): el conjunto de datos del formulario se agrega al final del URI especificado por el atributo action, con un signo de interrogación (?) como separador. Es el nuevo URI el que se envía al agente procesador. El método get debería usarse cuando el formulario no tiene efectos secundarios, como las búsquedas en bases de datos. • "post": el conjunto de datos del formulario se incluye en el cuerpo del formulario y se envía en dos procedimientos: primero contacta la URL definida por el atributo action, y luego si el contacto es exitoso, transmite los contenidos a dicha URL. Debe usarse este método si el servicio asociado con el procesamiento de un formulario causa efectos secundario (por ejemplo, si el formulario modifica una base de datos o la suscripción a un servicio). El método get restringe los valores del conjunto de datos del formulario a caracteres ASCII. Sólo el método post (con enctype ="multipart/form-data") cubre el conjunto de caracteres [ISO10646] completo.		

enctype <code>enctype="application/x-www-form-urlencoded "</code> <code>enctype="multipart/form-data "</code>	<p>Este atributo especifica el tipo de contenido (MIME type) usado para enviar el formulario al servidor (cuando el valor del atributo method sea "post"). El valor por defecto de este atributo es "application/x-www-form-urlencoded". El valor "multipart/form-data" debería usarse en combinación con el elemento INPUT, <code>type="file"</code>.</p>
accept-charset <code>accept-charset="unknown"</code>	<p>Este atributo especifica la lista de codificaciones de caracteres para los datos introducidos que son aceptadas por el servidor que procesa este formulario. El valor es una lista de valores de codificaciones de caracteres separadas por espacios y/o comas. En esta lista el servidor es capaz de aceptar cualquier codificación de caracteres individual por entidad recibida. El valor por defecto de este atributo es la cadena reservada "UNKNOWN" ("desconocido"). Los agentes de usuario pueden interpretar este valor como la codificación de caracteres que fue usada para transmitir el documento que contiene este elemento FORM.</p>
accept <code>accept="image/gif"</code>	<p>Este atributo especifica una lista de tipos de contenido separados por comas que un servidor procesador de formularios manejará correctamente. Los agentes de usuario pueden utilizar esta información para filtrar ficheros no conformes cuando pidan al usuario seleccionar ficheros para enviar al servidor (véase el elemento INPUT cuando <code>type="file"</code>).</p>
name <code>name="nombre"</code>	<p>Este atributo da nombre al elemento de modo que se pueda hacer referencia a él desde hojas de estilo o scripts. Este nombre debe ser único y no debe reutilizarse. Nota: Este atributo ha sido incluido por motivos de compatibilidad con versiones anteriores. Las aplicaciones deberían usar el atributo id para identificar elementos.</p>

Ejemplo

En el siguiente ejemplo se muestra un formulario que será procesado por el programa "contratar" cuando sea enviado y que será enviado a dicho programa usando el método HTTP `post`.

```
<form action="http://www.dominio.com/contratar" method="post">
... texto y controles del formulario ...
</form>
```

ELEMENTO INPUT

Sintaxis: (etiqueta inicial obligatoria y etiqueta final prohibida)

```
<input />
```

El elemento **INPUT** es utilizado para crear controles de formulario individuales (campos). Control es un término técnico que hace referencia a los elementos (botones, casillas de verificación, textareas, etc.) que pueden utilizarse dentro de un formulario para obtener información.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir de teclado: accesskey, tabindex eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario: eventos de formularios: onchange, onselect eventos de foco: onfocus, onblur estado del control: disabled ("disabled"), readonly ("readonly") otros: name (CDATA), value (CDATA), disabled ("disabled"), size (CDATA), type ("text" "password" "checkbox" "radio" "submit" "reset" "file" "hidden" "image" "button"), maxlength (Number), src (URI), alt (CDATA), usemap (URI), accept (ContentTypes)	align (ImgAlign")	Empty	
Desarrollo de atributos			
type type="text"	Especifica el tipo de control a crear , el valor por defecto para este atributo es "text", pero existen también: password, checkbox, radio, submit, reset, file, hidden, image y button.		
name name="nombre"	Este atributo asigna el nombre de control , el cuál debe ser único y no debe reutilizarse.		
value value="12/05/2007"	Este atributo especifica el valor inicial del control . Es opcional excepto cuando el atributo type tenga el valor "radio" o "checkbox".		
size size="50"	Dice el ancho inicial del control , que viene dado en píxeles excepto cuando el atributo type tiene el valor "text" o "password", en cuyo caso se refiere al número (entero) de caracteres.		
maxlength maxlength="200"	Cuando el atributo type tiene el valor "text" o "password", este atributo especifica el número máximo de caracteres que puede introducir el usuario. Este número puede exceder del especificado por size , en cuyo caso el agente de usuario debería ofrecer un mecanismo de desplazamiento. El valor por defecto para este atributo es un número ilimitado.		
checked checked="checked"	Cuando el atributo type tiene el valor "radio" o "checkbox", este atributo booleano especifica que el botón está marcado ("on").		
src src="URI"	Cuando el atributo type tiene el valor "image", este atributo especifica la localización de la imagen que debe usarse para decorar el botón gráfico de envío.		

Tipos de controles

Existen diez tipos de controles que pueden crearse aplicando el atributo **type** al elemento **INPUT**: button, checkbox, file, hidden, image, password, radio, reset, submit, text.

Todos requieren el atributo **name** menos submit y reset.

🔗 Ver los tipos de controles a crear con INPUT (CD > ejemplos > xhtml > forms > input.html)

type="button"

Crea un botón **pulsador (push button) rectangular** que puede ser clickeado cuando se desea que ocurra una acción. Si se desea enviar o resetear el formulario debería usarse los controles con type="submit" o

`type="reset"` respectivamente.

Se requiere un valor para el atributo **name**. Se puede utilizar el atributo **value** para asignar un texto que será mostrado en la superficie del botón. Dicho texto establece el tamaño mínimo, pero se puede establecer otro tamaño con el atributo **size** o con hojas de estilo. El color por defecto es gris, pero puede cambiarse utilizando hojas de estilo.

```
<input type="button" name="btn1" value="Aceptar" />
```

`type="checkbox"`

Crea un **botón cuadrado (casilla de verificación)** que puede ser seleccionado o deseleccionado con un clic del mouse. Cuando el checkbox está seleccionado, un signo de chequeado aparece dentro del cuadrado.

Se requiere un valor para el atributo **name**. Se puede utilizar el atributo **value** para asignar un valor asociado con el checkbox y el atributo **checked** para preseleccionar una casilla. Cuando una casilla es seleccionada, el valor es guardado como parte de la información que será procesada al enviar el formulario (las casillas no seleccionadas no son guardadas).

```
<input type="checkbox" name="ckbx1" value="true" />
```

`type="file"`

Se utiliza para adjuntar un archivo a un formulario a través de un control de selección de fichero (file select). Cuando el formulario es enviado, el archivo es pasado con el resto del contenido. Este control muestra una ventana **INPUT** y un botón de navegación. Cuando se hace click en este botón, el usuario puede buscar en sus directorios el archivo deseado. Una vez encontrado, con un doble clic se puede seleccionar dicho archivo. Como alternativa, en lugar de buscar el archivo, se puede ingresar la ruta y el nombre en la ventana **INPUT**. Se requiere el atributo **name** y no se utiliza el atributo **value**.

Para utilizar este tipo de **INPUT**, se debe colocar los atributos **enctype** y **method** al elemento **FORM** tal como se muestra a continuación:

```
<form method="post" enctype="multipart/form-data" action="procesar.php">  
<input type="file" name="file1" />
```

`type="hidden"`

Crea un control oculto y se utiliza para agregar contenido al formulario que no puede ser visualizado o cambiado por el usuario. Cuando el formulario es enviado, el contenido oculto se envía junto con el resto del contenido del formulario. Dicho contenido es una dupla nombre/valor. Se debe proveer un **name**, y un **value**, que puede ser cualquier información textual o numérica.

```
<input type="hidden" name="prueba" value="6467_asjd" />
```

`type="image"`

Crea un botón de envío gráfico y permite utilizar una imagen como equivalente de un botón del tipo submit. De hecho, cuando se clickee la imagen, el formulario será enviado.

También permite mostrar un mapa de imagen sensible al mouse dentro del formulario (para ello el navegador debe capturar las coordenadas **x** e **y** del mouse. Cuando se utiliza un dispositivo apuntador para hacer clic sobre la imagen, se envían al servidor el formulario y las coordenadas en que se pulsó el dispositivo. El valor **x** se mide en píxeles desde la izquierda de la imagen, y el valor **y** en píxeles desde la parte superior de la imagen. Los datos enviados incluyen **name.x=x-value** y **name.y=y-value** donde "**name**" es el valor del atributo **name**, y **x-value** e **y-value** son las coordenadas **x** e **y**, respectivamente.

Si el servidor realiza acciones diferentes dependiendo del lugar en que se pulsó el dispositivo, los usuarios de navegadores no gráficos estarán en desventaja. Por esta razón, los autores deberían considerar otras alternativas:

- Usar varios botones de envío (cada uno con su imagen) en lugar de un solo botón gráfico de envío.
- Usar un mapa de imágenes en el lado del cliente junto con scripts.

Se debe utilizar el atributo **src** para indicar la ubicación del archivo de imagen, y el atributo **alt** para proveer un texto a los navegadores que no muestran imágenes.

```
<input type="image" src="/images/submit.gif" onclick="submitform()"
name="submitgif" />
```

type="password"

Muestra un control de entrada de texto de una línea que permite ingresar contraseñas enmascaradas, con lo cual al ingresar una clave, cada carácter es reemplazado por un asterisco (*). El valor actual es el texto introducido por el usuario, no el texto representado por el agente de usuario.

El tamaño por defecto depende del navegador, y si bien varía entre 20-30 caracteres, el mismo puede ser cambiado con el atributo **size** u hojas de estilo. Se puede utilizar el atributo **maxlength** para establecer el número máximo de caracteres que aceptará el control.

Este método no implica seguridad, ya que si bien la clave no es visible en la pantalla, cuando el formulario es enviado, la clave pasa sin encriptación junto con los demás datos del formulario.

```
<input type="password" value="pass" />
```

type="radio"

Se utiliza para crear un **pequeño botón circular (radiobotón)** que puede ser seleccionado o deseleccionado con un clic del mouse. Cuando el mismo está seleccionado, un punto negro aparece en el centro del círculo.

Esto permite elegir entre varias opciones, pero sólo cuando una opción puede ser seleccionada. Cada grupo relacionado de botones de radio deben contener el mismo nombre, pero un valor diferente. Cuando un radio es seleccionado, sólo el valor de dicho radio será enviado con los contenidos del formulario. No es posible cambiar el tamaño o color de estos botones. Se puede utilizar el atributo **checked** para preseleccionar un botón.

```
<input type="radio" name="radio1" value="one" checked="checked" />
<input type="radio" name="radio1" value="two" />
```

type="reset"

Crea un **botón rectangular de reinicialización** que al ser clickeado limpia todos los campos del formulario, mostrándolos en su presentación original.

Generalmente es gris y con el texto "reset", sin embargo, el color se puede cambiar con hojas de estilo y el texto con el atributo **value**. Dicho texto indica el tamaño mínimo que puede ser reajustado con hojas de estilo o el atributo **size**.

```
<input type="reset" value="Clear The Form" name="reset1" />
```

type="submit"

Crea un botón rectangular que puede ser clickeado para **enviar el contenido de un formulario** para ser procesado.

El formulario puede enviarse a la misma página o a otra URL, lo cual está especificado en el elemento **FORM** a través del atributo **action**. Generalmente es gris y con el texto "Submit Query", sin embargo, el color se puede cambiar con hojas de estilo y el texto con el atributo **value**. Dicho texto indica el tamaño mínimo que puede ser reajustado con hojas de estilo o el atributo **size**. Si bien el atributo **name** no se requiere, es común aplicar un nombre único a cada control dentro de un formulario.

```
<input type="submit" name="submit" onclick="submitform()" value="Submit" />
```

type="text"

Crea un control de entrada de texto de una línea donde el usuario puede tipear la información solicitada (una ciudad, teléfono, email). Si se requiere que la información tenga más de una línea, entonces es conveniente el uso del elemento **TEXTAREA**. El tamaño por defecto depende del navegador, y ronda los 20-30 caracteres, sin embargo se recomienda establecer el tamaño a través de hojas de estilo o el atributo **size** para que la apariencia sea independiente del navegador. El atributo **maxlength** puede utilizarse para establecer el número máximo de caracteres que aceptará el control. El atributo **value** permitirá de manera opcional, asignar un texto que se mostrará dentro del control.

```
<input type="text" name="text1" size="50" />
```

Ejemplos

Solicitud de datos personales

CÓDIGO

El siguiente ejemplo define un formulario que permite introducir el nombre, apellido, email y sexo. Cuando se activa el botón de envío, el formulario es enviado al programa especificado por el atributo **action**.

```
<form action="http://www.dominio.com/contratar" method="post">
  Nombre: <input type="text" name="nombre" /><br />
  Apellido: <input type="text" name="apellido" /><br />
  Email: <input type="text" name="email" /><br />
  <input type="radio" name="sexo" value="Varón" /> Varón<br />
  <input type="radio" name="sexo" value="Mujer" /> Mujer<br />
  <input type="submit" value="Enviar" /> <input type="reset" />
</form>
```

RESULTADO

Esto podría representarse como sigue:

Envío de ficheros

CÓDIGO

Este ejemplo muestra cómo pueden enviarse los contenidos de un fichero especificado por el usuario con un formulario. Al especificar para **enctype** el valor **"multipart/form-data"**, los contenidos de cada fichero se empaquetarán para su envío en una sección separada de un documento multiparte:

```
<form action="http://www.dominio.com/archivos" enctype="multipart/form-data"
method="post">
  Nombre: <input type="text" name="nombre" />
  Archivos a enviar: <input type="file" name="archivos" />
</form>
```

ELEMENTO BUTTON

Sintaxis: (etiqueta inicial y final obligatoria)

<button>...</button>

El elemento **BUTTON** es utilizado para crear un botón de un formulario que puede tener contenido, consistir en una imagen y dar la ilusión de 3 dimensiones. Un botón creado con la etiqueta **INPUT** y `type="image"` no puede tener contenido ni apariencia de tres dimensiones.

Cualquier texto o imagen colocada dentro de la etiqueta de apertura y cierre de este elemento será mostrada en el botón. El texto puede incluir casi todas las etiquetas HTML, pero no se pueden utilizar mapas de imagen con los elementos **MAP** y **AREA**.

Los botones creados con el elemento **BUTTON** funcionan igual que los creados con **INPUT**, pero ofrecen posibilidades más ricas de representación, ya que el elemento **BUTTON** puede tener contenido. Por ejemplo, un elemento **BUTTON** que contenga una imagen se parece y funciona como un elemento **INPUT** cuyo atributo `type` es igual a `"image"`, pero el tipo de elemento **BUTTON** permite un contenido.

Los agentes de usuario visuales pueden representar los botones **BUTTON** con un relieve y un movimiento arriba/abajo al pulsarlos, mientras que representan los botones **INPUT** como imágenes "planas".

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir de teclado: accesskey, tabindex eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario: eventos de foco: onfocus, onblur estado del control: disabled ("disabled") otro: <u>name</u> (CDATA), <u>value</u> (CDATA), <u>type</u> ("button" "submit"* "reset")		Flow (acepta elementos en línea y en bloque)	A FORM INPUT SELECT TEXTAREA LABEL BUTTON FIELDSET IFRAME
Desarrollo de atributos			
name <code>name="nombre"</code>	Este atributo asigna el nombre de control.		
value <code>value="valorinicial"</code>	Este atributo asigna al botón su valor inicial.		
type <code>type="submit"</code> <code>type="reset"</code> <code>type="button"</code>	Este atributo declara el tipo del botón y los valores posibles son: <ul style="list-style-type: none"> submit: Es el valor por defecto, y crea un botón de envío (submit button). reset: Crea un botón de reinicialización (reset button). button: Crea un botón pulsador (push button). 		

Ejemplo

CÓDIGO

El siguiente ejemplo es en base a uno anterior, pero creando los botones de envío y de reinicialización con **BUTTON** en lugar de con **INPUT**. Los botones contienen imágenes sacadas de elementos **IMG** las cuáles deben tener un texto alternativo.

```
<form action="http://www.dominio.com/contratar" method="post">
  Nombre: <input type="text" name="nombre" /><br />
  Apellido: <input type="text" name="apellido" /><br />
  Email: <input type="text" name="email" /><br />
  <input type="radio" name="sexo" value="Varón" /> Varón<br />
  <input type="radio" name="sexo" value="Mujer" /> Mujer<br />
  <button name="enviar" value="enviar" type="submit">Enviar</button>
  <button name="reiniciar" type="reset">Limpiar</button>
</form>
```

ELEMENTO SELECT

Sintaxis: (etiqueta inicial y final obligatoria)

```
<select>...</select>
```

El elemento **SELECT** crea un menú. Cada opción ofrecida por el menú se representa por un elemento **OPTION**, y **SELECT** debe contener al menos un elemento **OPTION**. El elemento **OPTGROUP** permite agrupar opciones lógicamente, lo que es útil cuando existe una larga lista de opciones.

Se utiliza para delimitar el inicio y final de un menú de varias opciones. Por defecto se crea un control de una sola línea con un botón a su derecha, que al ser clickeado despliega las otras opciones disponibles. Se puede seleccionar un ítem de la lista haciendo clic sobre el mismo. Cuando un ítem es seleccionado, su contenido es enviado con el resto del formulario.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir de teclado: accesskey, tabindex eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario: eventos de formularios: onchange eventos de foco: onfocus, onblur estado del control: disabled ("disabled") otro: name (CDATA), size (Number), multiple ("multiple")		(OPTGROUP OPTION) + (uno o más elementos OPTGROUP y OPTION)	
Desarrollo de atributos			
name name="nombre"	Este atributo asigna el nombre de control.		
size size="50"	Los elementos SELECT se pueden presentar como una lista ("list box") o como un menú desplegable ("drop-down menu"). Si el elemento se presenta como una lista con desplazamiento ("scrolled list box"), este atributo especifica el número de filas de la lista que deberían ser visibles al mismo tiempo.		

Atributos aceptados	Modelo de contenido
multiple <i>multiple="multiple"</i>	Si está activado, este atributo booleano permite selecciones múltiples. Si no está activado, el elemento SELECT sólo permite selecciones simples.

Nota: Ver ejemplos de [SELECT](#), [OPTGROUP](#) y [OPTION](#).

ELEMENTO OPTGROUP

Sintaxis: (etiqueta inicial y final obligatoria)

```
<optgroup>...</optgroup>
```

El elemento **OPTGROUP** permite agrupar ítems relacionados en un menú. Un ítem sólo puede ser agregado al grupo a través del elemento **OPTION**. Tanto **OPTGROUP** como **OPTION** deben crearse dentro de las etiquetas de apertura y cierre del elemento **SELECT**. **OPTGROUP** no puede anidarse.

Atributos aceptados	Modelo de contenido		
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir de teclado: accesskey, tabindex eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario: eventos de formularios: onchange eventos de foco: onfocus, onblur estado del control: disabled ("disabled") otros: name (CDATA), size (Number), multiple ("multiple")	disabled ("disabled"), label (Text)	(OPTION) + (una o más etiquetas OPTION)	
Desarrollo de atributos			
label <i>label="nombre"</i>	Este atributo especifica el rótulo del grupo de opciones.		

Nota: Ver ejemplos de [SELECT](#), [OPTGROUP](#) y [OPTION](#).

ELEMENTO OPTION

Sintaxis: (etiqueta inicial y final obligatoria)

```
<option>...</option>
```

El elemento **OPTION** se utiliza para insertar un ítem en un menú. Se puede agregar cualquier cantidad de estos elementos, tantos como ítems se necesiten. Por defecto, cuando un ítem es seleccionado, el valor enviado con el formulario es el especificado luego de la etiqueta de apertura del elemento. Sin embargo se puede especificar otro valor a través del atributo **value**.

Este elemento debe aparecer entre la etiqueta de apertura y cierre de **SELECT**, y el orden de los elementos **OPTION** indicará el orden por defecto de la lista. Sin embargo, se puede utilizar el atributo **selected** para especificar cuál de los ítems aparecerá al principio de la lista.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout estado del control: disabled ("disabled") otros: <code>selected</code> ("selected"), <code>label</code> (Text), <code>value</code> (CDATA)		PCDATA	
Desarrollo de atributos			
selected <i>selected="selected"</i>	Si está establecido, este atributo booleano especifica que esta opción está preseleccionada. Pueden preseleccionarse cero o más opciones. Como el comportamiento de los agentes de usuario difiere frente a la falta de un elemento preseleccionado, los autores deberían asegurarse de que todos los menús incluyen un OPTION preseleccionado por defecto: <ul style="list-style-type: none"> • Si un elemento OPTION tiene el atributo selected activado, debería estar preseleccionado. • Si el elemento SELECT tiene el atributo multiple activado y hay más de un elemento OPTION que tenga el atributo selected activado, deberían estar todos preseleccionados. • Se considera un error que haya más de un elemento OPTION con el atributo selected activado y que el elemento SELECT no tenga el atributo multiple establecido. 		
value <i>value="valorinicial"</i>	Este atributo especifica el valor inicial del control. Si este atributo no está establecido, el valor inicial es igual a los contenidos del elemento OPTION .		
label <i>label="nombre"</i>	Este atributo permite a los autores especificar un rótulo para la opción, más corto que el contenido del elemento OPTION . Cuando está especificado, los agentes de usuario deberían usar como rótulo de la opción el valor de este atributo en lugar del contenido del elemento OPTION .		

Ejemplos

Utilización de select y option

CÓDIGO

Con el elemento **SELECT** creamos un menú que le permite al usuario seleccionar qué archivos desea descargar. Los dos primeros archivos están preseleccionados pero pueden ser deseleccionados por el usuario. El atributo **size** dice que el menú sólo tiene cuatro filas, aunque el usuario pueda elegir entre siete opciones.

```
<form action="http://www.dominio.com/archivo" method="post">
  <select multiple size="4" name="descargar_archivo">
    <option selected="selected" value="Archivo_1">Archivo a</option>
    <option selected="selected" value="Archivo_2">Archivo b</option>
    <option>Archivo c</option>
    <option>Archivo d</option>
    <option>Archivo e</option>
    <option>Archivo f</option>
    <option>Archivo g</option>
  </select>
  <input type="submit" value="Enviar" /><input type="reset" />
</form>
```

Solamente las opciones seleccionadas tendrán éxito (usando el nombre de control "descargar_archivo"). Cuando no haya opciones seleccionadas, el control no tendrá éxito, y ni el nombre ni ninguno de los valores

se enviarán al servidor cuando se envíe el formulario. El atributo **value**, cuando está establecido, determina el valor inicial del control, que de otro modo es el contenido del elemento.

Utilización de select, optgroup y option

CÓDIGO

En otro ejemplo usamos el elemento **OPTGROUP** para agrupar opciones:

```
<form action="http://www.dominio.com/archivo" method="post">
  <select name="descargar_archivo">
    <option selected="selected" label="ninguno" value="ninguno">Ninguno</option>
    <optgroup label="Fedora Core">
      <option label="4" value="FC_4">Fedora Core 4</option>
      <option label="5" value="FC_5">Fedora Core 5</option>
      <option label="6" value="FC_6">Fedora Core 6</option>
    </optgroup>
    <optgroup label="Red Hat Enterprise">
      <option label="4" value="RHE_4">Red Hat Enterprise 4</option>
      <option label="5" value="RHE_5">Red Hat Enterprise 5</option>
    </optgroup>
  </select>
</form>
```

El código anterior representa el siguiente agrupamiento:

```
Ninguno
Fedora Core
  4
  5
  6
Red Hat Enterprise
  4
  5
```

Los agentes de usuario visuales pueden permitir a los usuarios seleccionar de entre grupos de opciones a través de un menú jerárquico o de algún otro mecanismo que refleje la estructura de las opciones.

ELEMENTO TEXTAREA

Sintaxis: (etiqueta inicial y final obligatoria)

```
<textarea>...</textarea>
```

El elemento **TEXTAREA** crea un control de entrada de texto multilínea. Los agentes de usuario deberían usar los contenidos de este elemento como valor inicial del control y representar este texto inicialmente.

Este elemento debe utilizarse cuando se requiere el ingreso de información en más de una línea, de lo contrario conviene el uso del elemento input con `type="text"`. Cualquier texto o código que aparezca entre las etiquetas de apertura y cierre de **TEXTAREA**, será mostrado dentro de la ventana del **TEXTAREA**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir de teclado: accesskey, tabindex eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario: eventos de formularios: onchange, onselect eventos de foco: onfocus, onblur estado del control: disabled ("disabled"), readonly ("readonly") otro: <u>name</u> (CDATA), <u>rows</u> (Number), <u>cols</u> (Number)		PCDATA	
Desarrollo de atributos			
name <i>name="nombre"</i>	Este atributo asigna el nombre de control.		
rows <i>rows="5"</i>	Este atributo especifica el número de líneas de texto visibles. Los usuarios deberían poder introducir un número mayor de líneas, por lo que los agentes de usuario deberían proporcionar medios para desplazar los contenidos del control cuando los contenidos se extiendan más allá del área visible.		
cols <i>cols="50"</i>	Este atributo especifica el ancho visible en caracteres de anchura media. Los usuarios deberían poder introducir líneas de mayor longitud, por lo que los agentes de usuario deberían proporcionar medios para desplazar los contenidos del control cuando los contenidos se extiendan más allá del área visible. Los agentes del usuario pueden partir las líneas de texto visible para que las líneas largas puedan verse sin necesidad de desplazarlas.		

Ejemplo

CÓDIGO

Este ejemplo crea un control **TEXTAREA** de 10 filas por 95 columnas que contiene inicialmente dos líneas de texto. El elemento **TEXTAREA** va seguido de botones de envío y reinicialización.

```
<form action="http://www.dominio.com/politicasdeuso" method="post">
  <textarea name="eltexto" rows="10" cols="95">
    Texto inicial, primera línea.
    Texto inicial, segunda línea
  </textarea>
  <input type="submit" value="Enviar" /><INPUT type="reset" />
</form>
```

Estableciendo el atributo **readonly** los autores pueden mostrar texto no modificable en un elemento **TEXTAREA**. Esto no es lo mismo que usar código de texto estándar, ya que el valor del **TEXTAREA** se envía con el formulario.

ELEMENTO LABEL

Sintaxis: (etiqueta inicial y final obligatoria)

```
<label>...</label>
```

A algunos controles de formulario se les asocian rótulos automáticamente (botones pulsadores), en ellos los agentes

de usuario deberían utilizar el valor del atributo **value** como texto del rótulo.

El elemento **LABEL** se utiliza para asociar etiquetas de texto a elementos específicos (controles) de un formulario, especialmente a controles que no tienen rótulos implícitos (campos de texto, casillas de verificación y radiobotones, y menús). Cada elemento **LABEL** se asocia exactamente con un control de formulario. Esto se hace utilizando un mismo nombre o valor en el atributo **for** del elemento **LABEL** y el atributo **id** del control del formulario. Se puede asignar más de un **LABEL** a un mismo elemento.

Cuando el foco se dirige hacia un elemento **LABEL**, éste pasa el foco a su control asociado.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir de teclado: accesskey eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario: eventos de foco: onfocus, onblur otros: for (IDREF)		Inline (acepta elementos en línea)	LABEL
Desarrollo de atributos			
for <i>for="nombre"</i>		Este atributo asocia explícitamente el rótulo definido con otro control. Cuando está presente, el valor de este atributo debe ser el mismo que el valor del atributo id de algún otro control del mismo documento. Cuando no está presente, el rótulo definido se asocia con los contenidos del elemento. Se puede asociar más de un LABEL con el mismo control creando múltiples referencias a través del atributo for .	

Ejemplo

De forma sencilla

CÓDIGO

Entonces, podemos alinear un control de entrada de texto con su rótulo de la siguiente forma:

```
<label for="nombre">Nombre</label>
<input type="text" id="nombre" />
```

Aplicación de label a un control de formulario

CÓDIGO

Este ejemplo extiende el formulario de un ejemplo previo para que incluya elementos **LABEL**:

```
<form action="http://www.dominio.com/contratar" method="post">
  <label for="nombre">Nombre:</label>
  <input type="text" id="nombre" /><br />
  <label for="apellido">Apellido:</label>
  <input type="text" id="apellido" /><br />
  <label for="email">Email:</label>
  <input type="text" id="email" /><br />
  <input type="radio" name="sexo" value="Varón" /> Varón<br />
  <input type="radio" name="sexo" value="Mujer" /> Mujer<br />
  <input type="submit" value="Enviar" /> <input type="reset" />
</form>
```

Asociación implícita de un rótulo

Para asociar implícitamente un rótulo con otro control, el elemento de control (por ejemplo, un **INPUT**) debe estar dentro de los contenidos del elemento **LABEL**. En ese caso, el **LABEL** sólo puede contener un elemento de control. El rótulo en sí puede colocarse antes o después del control asociado.

CÓDIGO

En este ejemplo, asociamos implícitamente dos rótulos a dos controles de entrada de texto:

```
<form action="..." method="post">
  <label>
    Nombre
    <input type="text" name="nombre" />
  </label>
  <label>
    <input type="text" name="apellido" />
    Apellido
  </label>
</form>
```

Obsérvese que esta técnica no puede utilizarse cuando se usa una tabla para fijar la disposición de los elementos, con el rótulo en una celda y su control asociado en otra celda.

ELEMENTO FIELDSET

Sintaxis: (etiqueta inicial y final obligatoria)

```
<fieldset>...</fieldset>
```

El elemento **FIELDSET** (grupo de campos) permite a los autores agrupar temáticamente controles y rótulos relacionados. Gracias al agrupamiento de controles es más fácil para los usuarios entender su propósito y al mismo tiempo se facilita la navegación con agentes de usuario visuales y la navegación por voz para agentes de usuario basados en voz. El uso correcto de este elemento hace los documentos más accesibles.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario:		(LEGEND Flow) * (0 o más LEGEND y elementos en línea y en bloque)	

Nota: Ver ejemplos de **FIELDSET** y **LEGEND**.

ELEMENTO LEGEND

Sintaxis: (etiqueta inicial y final obligatoria)

```
<legend>...</legend>
```

El elemento **LEGEND** permite a los autores asignar un título a un **FIELDSET** y por lo tanto, a los grupos de elementos que este contiene. La leyenda mejora la accesibilidad cuando el **FIELDSET** no se representa visualmente.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir de teclado: accesskey eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario:	align ("left" "right" "top" "bottom")	Inline (acepta elementos en línea)	

Ejemplo

Aplicación de fieldset y legend

CÓDIGO

En el siguiente formulario se solicita información de la empresa, personal y del servicio a contratar, y cada sección tiene los controles necesarios para ingresar la información:

```
<form action="http://www.dominio.com/contratar" method="post">
<fieldset>
<legend>Información de la Empresa</legend>
  <label for="razonsocial">Razón Social</label>
  <input value="" name="razonsocial" id="razonsocial" size="50" maxlength="100"
    tabindex="1" type="text"><br />
  <label for="cuit">CUIT</label>
  <input value="" name="cuit" id="cuit" size="50" maxlength="11" tabindex="2"
    type="text"><br />
  <label for="ingresosbrutos">Ingresos Brutos</label>
  <input value="" name="ingresosbrutos" id="ingresosbrutos" size="50"
    maxlength="10" tabindex="3" type="text"><br />
  <label for="iva">Condición de IVA</label>
  <select name="iva" size="1" id="iva" tabindex="4">
    <option selected="selected">Seleccionar</option>
    <option>Responsable inscripto</option>
    <option>Responsable no inscripto</option>
    <option>Excento</option>
  </select><br />
</fieldset>

<fieldset>
<legend>Información Personal</legend>
  <label for="nombre">Nombre y Apellido</label>
  <input value="" name="nombre" id="nombre" size="50" maxlength="50" tabindex="5"
    type="text"><br />
  <label for="domicilio">Domicilio</label>
  <input value="" name="domicilio" id="domicilio" size="50" maxlength="100"
    tabindex="6" type="text"><br />
  <label for="telefono">Teléfono</label>
  <input value="" name="telefono" id="telefono" size="50" maxlength="25"
    tabindex="7" type="text"><br />
  <label for="email">EMail</label>
  <input value="" name="email" id="email" size="50" maxlength="150" tabindex="8"
    type="text"><br />
</fieldset>

<fieldset>
<legend>Plan a contratar</legend>
  <input name="servicio" value="55" id="plan55" checked="checked" tabindex="9"
    type="radio">
  <label for="55">Plan 50MB x 12 meses ($45 anual)</label><br />
  <input name="servicio" value="56" id="plan56" tabindex="10" type="radio">
```



```
<label for="56">Plan 100MB x 12 meses ($90 anual)</label><br />
</fieldset>
</form>
```

CONTROLES DESHABILITADOS Y DE SÓLO LECTURA

Cuando la entrada de datos por parte del usuario es indeseable o irrelevante, es posible deshabilitar (**disabled**) un control o convertirlo en un control de sólo lectura (**readonly**). Podríamos, por ejemplo, deshabilitar el botón de envío de un formulario si el usuario no ha introducido ciertos datos obligatorios.

disabled

Sintaxis: `disabled="disabled"`

Este **atributo booleano** deshabilita el control para la entrada de datos por parte del usuario. Cuando está establecido, el atributo **disabled** tiene los siguientes efectos:

- No permite dirigir el foco hacia los elementos con dicho atributo.
- En el orden de tabulación, los elementos con dicho atributo son saltados.
- Los controles deshabilitados no pueden tener éxito.

Los siguientes elementos soportan el atributo **disabled**: **BUTTON**, **INPUT**, **OPTGROUP**, **OPTION**, **SELECT** y **TEXTAREA**.

Este atributo se hereda, pero las declaraciones locales prevalecen sobre el valor heredado. El modo en que se representan los elementos deshabilitados depende del agente de usuario, algunos dibujan en gris los objetos de menú deshabilitados, los rótulos de los botones, etc.

En este ejemplo, el elemento **INPUT** está deshabilitado, no puede recibir datos del usuario, y su valor no se enviará con el formulario.

```
<input disabled="disabled" name="plan" value="Plan 50MB Promoción anual">
```

readonly

Sintaxis: `readonly="readonly"`

Este **atributo booleano** impide que haya cambios en el control. El atributo **readonly** especifica si el control puede ser modificado por el usuario y cuando está establecido:

- El foco puede dirigirse hacia elementos de sólo lectura, pero éstos no pueden ser modificados por el usuario.
- Los elementos de sólo lectura están incluidos en la navegación con tabulador.
- Los elementos de sólo lectura pueden tener éxito.

Los siguientes elementos soportan el atributo **readonly**: **INPUT** y **TEXTAREA**.

ENVÍO DE FORMULARIOS

Las siguientes secciones explican cómo envían los agentes de usuario los datos de los formularios a los agentes procesadores de formularios.

El URI que se construye cuando se envía un formulario puede utilizarse como un vínculo "estilo ancla" (Ej. en el atributo **href** del elemento **A**). La utilización del carácter "&" para separar los campos del formulario interfiere con su uso en los valores de los atributos SGML para delimitar referencias a entidades de caracteres.

Por ejemplo, para usar el URI siguiente como URI de un destino de vínculo:

```
http://www.dominio.com/prueba.php?x=1&y=2
```

Debe escribirse:

```
http://www.dominio.com/prueba.php?x=1&amp;y=2
```

Controles con éxito

Un control con éxito es "válido" para su envío y tienen su nombre de control emparejado con su valor actual como parte del conjunto de datos del formulario enviado. Un control con éxito debe estar definido dentro de un elemento **FORM** y debe tener un nombre de control.

Sin embargo:

- Los controles que están deshabilitados no pueden tener éxito.
- Si un formulario contiene más de un botón de envío, solamente el botón de envío activado tiene éxito.
- Todas las casillas de verificación (checkboxes) marcadas pueden tener éxito.
- Para los radiobotones (radio buttons) que compartan el mismo valor del atributo **name**, solamente el radiobotón marcado puede tener éxito.
- Para los menús, el nombre de control viene dado por un elemento **select** y los valores son proporcionados por elementos **OPTION**. Solamente las opciones seleccionadas pueden tener éxito. Cuando no haya opciones seleccionadas, el control no tiene éxito, y ni el nombre ni los valores se envían al servidor cuando se envía el formulario.
- El valor actual de un selector de archivos es una lista de uno o más nombres de archivos. Al enviar el formulario, los contenidos de cada archivo se envían con el resto de los datos del formulario. Los contenidos de los archivos se empaquetan de acuerdo con el tipo de contenido del formulario.
- El valor actual de un control tipo objeto está determinado por la implementación del objeto.

Si un control no tiene valor actual cuando se envía el formulario, los agentes de usuario no están obligados a tratarlo como un control con éxito.

Además de esto, los agentes de usuario no deberían considerar a los siguientes como controles con éxito:

- Botones de reinicialización (reset buttons).
- Elementos **OBJECT** cuyo atributo **declare** haya sido establecido.

Los controles ocultos y los controles que no sean representados debido a la configuración de una hoja de estilo pueden tener éxito. Por ejemplo:

```
<input type="password" style="display:none"  
      name="clave-oculta" value="clave" />
```

Procesamiento de los datos del formulario

Cuando el usuario envía un formulario, el agente de usuario lo procesa del siguiente modo:

- **Paso 1 - Identificar los controles con éxito.**
- **Paso 2 - Construir el conjunto de datos del formulario:** es una secuencia de parejas nombre de control/valor actual construida a partir de los elementos con éxito.
- **Paso 3 – Codificar el conjunto de datos del formulario:** El conjunto de datos del formulario se codifica a continuación de acuerdo con el tipo de contenido especificado por el atributo **enctype** del elemento **FORM**.
- **Paso 4 – Enviar el conjunto de datos del formulario codificado:** Los datos codificados se envían al agente procesador designado por el atributo **action** usando el protocolo especificado por el atributo

method. Los agentes de usuario deben soportar las convenciones establecidas en los siguientes casos:

- Si **method** es "get" y **action** es un URI HTTP, el agente toma el valor de **action**, le agrega un '?', y a continuación agrega el conjunto de datos del formulario, codificado según el tipo de contenido "application/x-www-form-urlencoded". A continuación el agente de usuario sigue este URI.
- Si **method** es "post" y **action** es un URI HTTP, el agente de usuario conduce una transacción HTTP "post" usando el valor del atributo **action** y un mensaje creado de acuerdo con el tipo de contenido especificado por el atributo **enctype**.

Tipos de contenido de formularios

El atributo **enctype** del elemento **form** especifica el tipo de contenido usado para codificar el conjunto de datos del formulario para su envío al servidor.

application/x-www-form-urlencoded

Este es el tipo de contenido por defecto y los formularios enviados con este tipo de contenido deben codificarse como sigue:

1. Los nombres de control y los valores se transforman en secuencias de escape. Los caracteres de espacio se sustituyen por '+', y los caracteres reservados se transforman en secuencias de escape: Los caracteres no alfanuméricos se reemplazan por '%HH', un signo de porcentaje y dos dígitos hexadecimales que representan el código ASCII del carácter. Los saltos de línea se representan como parejas "CR LF" (es decir, '%0D%0A').
2. Las parejas nombre de control/valor se enumeran según el orden en que aparecen en el documento. El nombre se separa del valor con un signo '=' y las parejas nombre/valor se separan entre sí con un signo '&'.

multipart/form-data

El tipo de contenido "application/x-www-form-urlencoded" no es eficiente para enviar grandes cantidades de datos binarios o textos que contenga caracteres no ASCII. Para enviar formularios que contengan ficheros, datos no ASCII, y datos binarios debería utilizarse el tipo de contenido "multipart/form-data".

Un mensaje "multipart/form-data" contiene una serie de partes, cada una de las cuales representa un control con éxito. Las partes se envían al agente procesador en el mismo orden en que aparecen los controles correspondientes en el flujo del documento.

Como con todos los tipos MIME multiparte, cada parte tiene un encabezado opcional "Content-Type" cuyo valor por defecto es "text/plain". Los agentes de usuario deberían proveer el encabezado "Content-Type", acompañado por un parámetro "charset".

14. Scripts



INTRODUCCIÓN A LOS SCRIPTS

Un script en el lado del cliente es un programa que puede acompañar a un documento XHTML o que puede estar incluido en él. El programa se ejecuta en la máquina del cliente cuando se carga el documento, o cuando se ejecuta alguna acción como activar un vínculo.

Los scripts ofrecen la posibilidad de extender los documentos XHTML de maneras activas e interactivas:

- Pueden evaluarse los scripts a medida que se carga el documento para modificar los contenidos dinámicamente.
- Los scripts pueden acompañar a un formulario para procesar los datos a medida que éstos se introducen. Permiten rellenar dinámicamente campos en base a los valores de otros campos y asegurarse que los datos introducidos concuerden con rangos de valores predeterminados, etc.
- Los scripts pueden ser llamados por eventos que afecten al documento, como la carga, la descarga, el movimiento del foco sobre los elementos, los movimientos del ratón, etc.
- Los scripts pueden ser vinculados a controles de formulario (Ej. botones) para producir elementos gráficos para la interfaz del usuario.

Hay dos tipos de scripts que los autores pueden asociar a un documento XHTML:

- Aquellos que **se ejecutan una sola vez** cuando el agente de usuario carga el documento (aparecen dentro de un elemento **SCRIPT**). Se puede incluir contenido alternativo por medio del elemento **NOSCRIPT**, para los agentes de usuario que no pueden ejecutar scripts.
- Aquellos que son **ejecutados cada vez que ocurre un determinado evento**. Estos scripts pueden ser asignados a varios elementos por medio de los atributos de **eventos intrínsecos**.

DISEÑO DE DOCUMENTOS PARA AGENTES DE USUARIO QUE SOPORTEN SCRIPTS

Elemento SCRIPT

Sintaxis: (etiqueta inicial y final obligatoria)

```
<script>...</script>
```

El elemento **SCRIPT** coloca un script dentro de un documento. Este elemento puede aparecer cualquier número de veces en el **HEAD** o en el **BODY** de un documento HTML.

El script puede estar definido dentro de los contenidos del elemento **SCRIPT** o en un fichero externo.

Si el atributo **src** no está establecido, los agentes de usuario deben interpretar que los contenidos del elemento son el script. Si **src** tiene un valor URI, los agentes de usuario no deben tener en cuenta los contenidos del elemento y deben obtener el script mediante el URI. El atributo **charset** se refiere a la codificación de caracteres del script designado por el atributo **src**; no afecta al contenido del elemento **SCRIPT**.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id otros: charset (Charset), type* (ContentType), src (URI), defer ("defer"), xml:space ("preserve")	language (CDATA)	PCDATA	
Desarrollo de atributos			
src <i>src="URI"</i>	Este atributo especifica la localización de un script externo.		
type <i>type="text/javascript"</i>	Este atributo obligatorio especifica el lenguaje de scripts de los contenidos del elemento y prevalece sobre el lenguaje de scripts por defecto. El lenguaje de scripts se especifica como un tipo de contenido (Ej. "text/javascript") y como no hay un valor por defecto, los autores deben proporcionar uno para este atributo.		
defer <i>defer="defer"</i>	Este atributo booleano indica al agente de usuario que el script no va a generar ningún contenido en el documento (Ej. en javascript, cuando no hubiera ningún "document.write") y por lo tanto el agente de usuario puede seguir analizando y representando.		

Especificación del lenguaje de scripts

Al no estar ligado el XHTML a un lenguaje de scripts específico, los autores de los documentos deben decir explícitamente a los agentes de usuario el lenguaje de cada script, lo que puede hacerse mediante una declaración por defecto o mediante una declaración local.

Declaración por defecto del lenguaje de scripts

El lenguaje de scripts por defecto de los scripts de un documento se puede especificar incluyendo la declaración **META** en el **HEAD**, donde "type" es un tipo de contenido que se refiere al lenguaje de scripts ("text/tcl", "text/javascript", "text/vbscript").

```
<meta http-equiv="Content-Script-Type" content="type" />
```

En ausencia de una declaración **META**, el valor por defecto puede ser establecido con un encabezado HTTP "Content-Script-Type".

Content-Script-Type: type

Los agentes de usuario deberían determinar el lenguaje de scripts por defecto de un documento de acuerdo con los siguientes pasos (ordenados de prioridad más alta a más baja):

1. Si alguna declaración **META** especifica el "Content-Script-Type", la última de ellas en el flujo de caracteres determina el lenguaje de scripts por defecto.
2. Si algún encabezado HTTP especifica el "Content-Script-Type", el último de ellos en el flujo de caracteres determina el lenguaje de scripts por defecto.

Los documentos que no especifiquen información relativa al lenguaje de scripts por defecto y que contengan elementos que especifiquen un script de evento intrínseco son incorrectos.

Declaración local del lenguaje de un script

Se debe especificar el atributo **type** de todos los elementos **SCRIPT** de un documento y el valor de **SCRIPT** en un elemento prevalece sobre el lenguaje de scripts por defecto de ese elemento.

Ejemplo: En el siguiente código:

- declaramos que el lenguaje de scripts por defecto es "text/tcl".
- incluimos un **script** en la cabecera localizado en un fichero externo, en lenguaje "text/vbscript".
- incluimos un **script** en el cuerpo, que contiene su propio script escrito en "text/javascript".

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba con SCRIPT</title>
  <meta http-equiv="Content-Script-Type" content="text/tcl" />
  <script type="text/vbscript" src="http://www.dominio.com/externo"></script>
</head>

<body>
  <script type="text/javascript">
    ...algo en JavaScript...
  </script>
</body>
</html>
```

Referencias a elementos HTML desde un script

Cada lenguaje de scripts tiene sus propias convenciones para referirse a objetos HTML desde dentro del script. Sin embargo, los scripts deberían hacer referencia a un elemento de acuerdo con su nombre asignado, siguiendo las reglas de precedencia cuando identifiquen un elemento: un atributo **name** prevalece sobre un atributo **id** si ambos están establecidos. En caso contrario, se puede usar uno u otro.

Eventos intrínsecos

Es posible asociar una acción con un cierto número de eventos que ocurren cuando un usuario interacciona con un agente de usuario. Cada uno de los "eventos intrínsecos" toma como valor un script que se ejecuta cada vez que el evento ocurre para ese elemento.

Los elementos de control tales como **INPUT**, **SELECT**, **BUTTON**, **TEXTAREA** y **LABEL** responden a ciertos eventos intrínsecos. Cuando estos elementos no aparecen dentro de un formulario, se pueden emplear para enriquecer la interfaz gráfica del usuario del documento. Por ejemplo, los autores pueden incluir botones en sus documentos que no envíen un formulario pero que puedan comunicarse con un servidor cuando son activados.

Ejemplo

CÓDIGO

“nombre” es un campo de texto obligatorio, entonces cuando un usuario intenta abandonar el campo, el evento **onblur** llama a una función JavaScript para confirmar que “nombre” tiene un valor aceptable.

```
<input name="nombre" onblur="validarnombre (this.value)">
```

Modificación dinámica de documentos

Los scripts que se ejecutan cuando un documento es cargado, pueden modificar los contenidos del documento dinámicamente. La capacidad de hacer esto depende del lenguaje de scripts en sí (Ej. la sentencia "document.write" en el modelo de objetos de HTML no está soportada por algunas marcas).

La modificación dinámica de un documento puede ser realizada de la siguiente manera:

1. Todos los elementos **script** se evalúan en orden a medida que el documento es cargado.
2. Todas las construcciones de scripts contenidas en un elemento **script** dado que generen datos CDATA SGML son evaluados. Su texto generado combinado se inserta en el documento sustituyendo al documento **script**.
3. Los datos CDATA generados son evaluados nuevamente.

Los documentos XHTML deben ser conformes con el DTD del XHTML, tanto antes como después del procesamiento de cualquiera de los elementos **script**.

Ejemplo

CÓDIGO

A continuación puede verse como un script puede modificar un documento dinámicamente:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba</title>
  <script type="text/javascript">
    document.write("<p><strong>;Hola!</strong></p>")
  </script>
</head>

<body>
</body>
</html>
```

Para tener el mismo efecto que:

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

<body>
  <p><strong>;Hola!</strong></p>
</body>
</html>
```

DISEÑO DE DOCUMENTOS PARA AGENTES DE USUARIO QUE NO SOPORTEN SCRIPTS

Los autores pueden crear documentos que funcionen para agentes de usuario que no soporten scripts y en las páginas siguientes veremos cuál es el procedimiento para lograrlo.

Elemento NOSCRIPT

Sintaxis: (etiqueta inicial y final obligatoria)

```
<noscript>...</noscript>
```

El elemento **NOSCRIPT** permite proporcionar contenido alternativo cuando un script no es ejecutado. El contenido de un elemento **NOSCRIPT** sólo debería ser representado por un agente de usuario capaz de reconocer scripts en los casos siguientes:

- El agente de usuario está configurado para no evaluar scripts.
- El agente de usuario no soporta un lenguaje de scripts invocado por un elemento **SCRIPT** anterior en el documento.

Atributos aceptados		Modelo de contenido	
XHTML Strict	Frameset, Transitional	Contiene a:	No acepta:
principales: id, class, style, title de lenguaje: lang, xml:lang, dir eventos de teclado: onkeypress, onkeydown, onkeyup eventos de mouse: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout eventos de formulario:		Block (acepta elementos en bloque)	

Ejemplo

CÓDIGO

En el ejemplo siguiente, un agente de usuario que ejecute el **SCRIPT** incluirá en el documento algunos datos creados dinámicamente. Si el agente de usuario no soporta scripts, el usuario aún podrá obtener los datos por medio de un vínculo.

```
<script type="text/javascript">
... script para insertar datos...
</script>
<noscript>
  <p>Acceder a los <a href="http://www.dominio.com/datos.html">datos.</a></p>
</noscript>
```

Ocultar datos de scripts a agentes de usuario

Los agentes de usuario que no reconocen el elemento **SCRIPT** probablemente representen los contenidos del elemento como texto. Algunos motores de scripts (incluyendo JavaScript, VBScript y Tcl) permiten que las sentencias de los scripts estén contenidas en un comentario SGML, de modo que aquellos agentes de usuario que no reconocen el elemento **SCRIPT** ignoren el comentario, y aquellos que si lo reconocen, lo ejecuten.

Otra solución al problema es mantener los scripts en documentos externos y hacer referencia a ellos con el atributo **src**.

Comentando scripts en Javascript

El motor de JavaScript permite que aparezca la cadena "`<!--`" al principio del elemento **SCRIPT**, e ignora el resto de los caracteres hasta el final de la línea. JavaScript interpreta "`//`" como el inicio de un comentario que se extiende hasta el final de la línea actual y es necesaria para ocultar la cadena "`-->`" al analizador JavaScript.

```
<script type="text/javascript">
<!-- para ocultar los contenidos del script a los navegadores viejos
    function cuadrado(i) {
        document.write("La llamada pasó ", i , " a la función.", "<BR>")
        return i * i
    }
    document.write("La función devolvió ", cuadrado(5), ".")
// dejar de ocultar contenidos a los navegadores viejos -->
</script>
```

Comentando scripts en VBScript

En VBScript, un carácter de comilla simple hace que el resto de la línea actual sea tratada como un comentario. Puede usarse por tanto para ocultar a VBScript la cadena "`-->`", por ejemplo:

```
<script type="text/vbscript">
<!--
    Sub blabla()
        ...
    End Sub
' -->
</script>
```

Comentando scripts en TCL

En Tcl, el carácter "`#`" comenta el resto de la línea:

```
<script type="text/tcl">
<!-- para ocultar los contenidos del script a los navegadores viejos
proc cuadrado {i} {
    document write "La llamada pasó $i a la función.<BR>"
    return [expr $i * $i]
}
document write "La función devolvió [cuadrado 5]."
# dejar de ocultar los contenidos a los navegadores viejos -->
</script>
```

Nota: Algunos navegadores cierran los comentarios al encontrar el primer carácter "`>`", de modo que para ocultar el contenido de los scripts de estos navegadores, se pueden invertir los operandos de los operadores relacionales y de desplazamiento (Ej. usar "`y < x`" en vez de "`x > y`") o se pueden usar caracteres de escape dependientes del lenguaje de scripts para "`>`".



Manual de CSS

CASCADING SYTLE SHEETS

CSS son las siglas de **Cascading StyleSheets** que en español significa Hojas de estilo en Cascada. La idea detrás de su desarrollo es separar la estructura de un documento de su presentación, ya que de hecho, las CSS son un mecanismo o lenguaje simple para agregar estilos (fuentes, colores, espaciado) a documentos Web estructurados (por ejemplo documentos HTML, XHTML, XML).

Dentro de un documento podemos diferencia entre el estilo lógico y el físico. El **estilo lógico** se refiere a la lógica o estructura del documento (cabeceras, párrafos, etc.) y no se preocupa de la apariencia final del mismo. Por el contrario, el **estilo físico** no se interesa por la estructura del documento, sino por la apariencia final: párrafos con un cierto tipo de letra, tablas con un determinado color de fondo...

El objetivo de las hojas de estilo es separar en un documento el estilo lógico (estructura) del físico (presentación), dejando este último en bloques de estilos separados de la estructura del documento. Es decir que las hojas de estilo son una especificación sobre los estilos físicos aplicables a un documento HTML.

1. Interpretación del manual de CSS



Como paso previo a la lectura del tema sobre CSS, se recomienda leer “[¿Cómo interpretar este manual?](#)” en donde encontrará términos utilizados asiduamente a lo largo de la redacción y otras consideraciones importantes a fin de comprender los textos siguientes.

DEFINICIÓN DE UNA PROPIEDAD CSS

Cada definición de una propiedad CSS comienza con un resumen de la información importante, que tiene el siguiente formato:

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'nombrepropiedad'	valores permitidos y sintaxis	valor inicial	elementos a los cuales se aplica esta propiedad	si es heredada	cómo se interpretan los valores de porcentaje	A qué medio se aplica	1, 2 o 3

Valor computado: cómo computar el valor computado

A continuación podrá ver una explicación de la información brindada por cada uno de estos puntos:

Valor

Indica el conjunto de los valores válidos para la propiedad. Los tipos de valor se pueden designar como:

1. valores de palabras clave que deben aparecer literalmente sin las comillas (Ej. `auto`, `disco`, `red`, etc.)
2. tipos de datos básicos que aparecen entre "<" y ">" (Ej. `<medida>`, `<porcentaje>`, etc.).
3. tipos que tienen la misma gama de valores que una propiedad que lleva el mismo nombre (Ej.

`<'border-width' > <'background-attachment'>`, etc.). En este caso, el nombre del tipo es el nombre de la propiedad entre comillas y entre "<" y ">" (Ej., `<'border-width'>`). Dicho tipo no incluye el valor 'heredado'.

4. no-terminales que no comparten el mismo nombre que una propiedad. En este caso, el nombre no-terminal aparece entre "<" y ">", como en `<border-width>`. Note la distinción entre la `<border-width>` y `<'border-width'>`; el último se define en los términos del anterior.

La raya vertical (/) y la coma (,) deben aparecer literalmente. Los valores pueden estar organizados de las siguientes maneras:

- Varias palabras yuxtapuestas significa que todas ellas deben aparecer, en el orden dado.
- Una barra (|) separa dos o más alternativas: exactamente una de ellas debe aparecer.
- Una barra doble (||) separa dos o más opciones: una o más de ellas debe aparecer, en cualquier orden.
- Los corchetes ([]) están para agrupar.

La yuxtaposición es más fuerte que la barra doble, y la barra doble es más fuerte que la barra. Así, las líneas siguientes son equivalentes:

```
a b | c || d e
[ a b ] | [ c || [ d e ] ]
```

Cada tipo, palabra clave o grupo encerrado entre corchetes puede ser seguido por uno de los siguientes modificadores:

- Un asterisco (*) indica que el tipo, la palabra o el grupo precedente aparece cero o más vez.
- Un signo más (+) indica que el tipo, la palabra o el grupo precedente aparece una o más veces.
- Un signo de interrogación (?) indica que el tipo, la palabra o el grupo precedente es opcional.
- Un par de números entre llaves ({ A,B }) indica que el tipo, la palabra o el grupo precedente aparece por lo menos A y a lo sumo B veces.

Los ejemplos siguientes ilustran diversos tipos de valor:

- Valor: `N | NW | NE`
- Valor: `[<medida> | thick | thin]{1,4}`
- Valor: `[<family-name> ,] * <family-name>`
- Valor: `<uri> ? <color> [/ <color>]`
- Valor: `<uri> || <color>`

La gramática permite espacios entre los símbolos y en muchos casos, los espacios entre símbolos pueden ser necesarios para distinguir unos de otros. Por ejemplo, el valor `'5em3em'` sería analizado como un solo símbolo con el número '5' y el identificador `'em3em'`, que sería una unidad no válida. En este caso, sería necesario un espacio antes del '2' para conseguir que lo tome como dos medidas `'1em'` y `'2em'`.

Inicial

Especifica el valor inicial de la propiedad. Si la propiedad se hereda, éste es el valor que se da al elemento raíz de la estructura del documento *.

Se aplica a

Registra los elementos a los cuales se aplica la propiedad. Si bien todos los elementos son aptos para tener todas las propiedades, algunas propiedades no tienen ningún efecto en ciertos elementos. Por ejemplo, la propiedad `'clear'` sólo afecta a los elementos a nivel de bloque.

Se hereda

Indica si el valor de la propiedad se hereda de un elemento antepasado.

Porcentaje

Indica cómo deben ser interpretados los porcentajes cuando aparecen en el valor de la propiedad. Si aquí aparece "N/A", significa que la propiedad no acepta porcentajes como valores.

Grupos de medios

Indica los grupos de medios a los cuales se aplica la propiedad.

Valor computado

Describe el valor computado para la propiedad. (Vea la sección sobre valores computados para ver su definición).

PROPIEDADES RESUMIDAS

Algunas propiedades son propiedades resumidas que permiten a los autores especificar los valores de varias propiedades con una sola propiedad.

Por ejemplo, la propiedad `'font'` es una propiedad resumida para definir `'font-style'`, `'font-variant'`, `'font-weight'`, `'font-size'`, `'line-height'` y `'font-family'` en un solo paso. Así también, `'margin'` permite congregarse en una sola propiedad, las cuatro siguientes: `'margin-top'`, `'margin-right'`, `'margin-bottom'` y `'margin-left'`.

Cuando algunos valores son omitidos en la fórmula resumida, a cada propiedad "ausente" le es asignado su valor inicial (ver Cascada).

Ejemplo

CÓDIGO EXPANDIDO

Las múltiples reglas de estilo del código expandido pueden ser reescritas en una sola propiedad tal como se muestra en el código resumido:

```
H1 {  
  font-weight: bold;  
  font-size: 12pt;  
  line-height: 14pt;  
  font-family: Helvetica;  
  font-variant: normal;  
  font-style: normal;  
}
```

CÓDIGO RESUMIDO

```
H1 { font: bold 12pt/14pt Helvetica }
```

Como `'font-variant'` y `'font-style'` toman sus valores iniciales no es necesario aclararlos.

2. Sintaxis y tipos de datos básicos



SINTAXIS

Estamentos

Una hoja de estilo de CSS consiste en una lista de **estamentos**. Hay dos clases de estamentos: **reglas-arroba** y **sistemas de reglas**. Puede haber espacios en blanco alrededor de los estamentos.

Reglas-arroba

Las Reglas-arroba comienzan con una clave-arroba (un carácter '@') seguida inmediatamente (sin mediación de un espacio en blanco) por un identificador (por ejemplo, '@import', '@page').

Una regla-arroba consiste en todo lo que hay hasta, e incluyendo, el punto y coma siguiente (;) o el siguiente bloque, cualquiera que sea el primero en aparecer.

```
@media print {  
  p {font-family:times,serif; font-size:10px}  
}  
@import "prueba.css";
```

Si una aplicación de usuario encuentra una regla-arroba desconocida debe ignorar el conjunto de la regla-arroba y continuar el análisis después de ella. Las aplicaciones de usuario deben ignorar cualquier regla '@import' que aparezca dentro de un bloque o que no preceda todas la reglas fijadas.

Si un programa de análisis encuentra esta hoja de estilo:

```
@import "prueba.css";  
h1 { color: red;}  
@import "listas.css";
```

Ignorará completamente la segunda regla-arroba ya que aparece luego de otra regla fijada, dejando la hoja de estilo como:

```
@import "prueba.css";  
h1 { color: red; }
```

En otro ejemplo, la segunda regla '@import' no es válida, porque aparece dentro de un bloque de '@media'.

```
@import "prueba.css";  
@media print {  
  @import "print-main.css";  
  body { font-size: 10pt; }  
}  
h1 {color: red; }
```

Sistemas de reglas

Un **sistema de las reglas**, también llamado **regla CSS** consiste en un **selector** seguido por un **bloque de declaraciones**. El bloque de declaraciones contiene una o más **declaraciones** dentro llaves. Y cada declaración está formada por una dupla **propiedad:valor**.

Entonces, frente a la siguiente estructura:

```
selector { propiedad: valor; }
```

Podríamos establecer la siguiente regla:

```
body { color: black; padding: 2px; }
```

Y así se identifica que:

- la **regla** es la línea completa: `body { color: black; padding: 2px; }`
- el **selector** es normalmente el elemento HTML que se desea definir: `body`
- el **bloque de declaraciones** es todo dentro e incluyendo las llaves: `{color: black; padding: 2px; }`
- las **declaraciones** son cada para dupla "propiedad:valor": `color: black` y `padding: 2px`
- las **propiedades** son el atributo que deseamos modificar: `color` y `padding`
- el **valor** es justamente el valor que se le aplica a las propiedades: `black` y `2px`.

Veamos a continuación una explicación más completa de cada una de las partes de una regla CSS:

Selectores

El **selector** consiste en todo lo que hay hasta (pero sin incluir) la primera llave izquierda ({}). Un selector siempre va junto con un bloque de declaraciones. Cuando una aplicación de usuario no puede analizar el selector (es decir, no es un CSS válido), debe ignorar también el bloque de declaraciones.

Por ejemplo, como el "&" no es un comando válido en un selector, la aplicación de usuario debe ignorar la segunda línea completa y no poner el color de **H3** como rojo:

```
h1, h2 {color: green }  
h3, h4 & h5 {color: red }  
h6 {color: black }
```

Los selectores y sus propiedades permiten hacer referencia a diferentes partes de un documento:

- Los elementos de la estructura del documento y ciertas relaciones entre ellos (Ver "[Selectores](#)"). Por ejemplo: **HEAD**, **BODY**, **P** **STRONG**.
- Los atributos de los elementos de la estructura del documento y los valores de esos atributos (Ver

“Selectores de atributos”. Por ejemplo: `h1[title]`

- Algunas partes del contenido de un elemento (Ver “Pseudo-elementos”) Por ejemplo: `p:first-letter`
- Los elementos de la estructura del documento cuando se encuentran en cierto estado (Ver “Pseudo-clases”). Por ejemplo: `a:link`
- Algunos aspectos del *lienzo** en el cual el documento será procesado.
- Alguna información del sistema (Ver “Interfaz de usuario”). Por ejemplo: `:link { cursor: hand; }`

Bloque de declaraciones

Un **bloque de declaraciones** comienza con una llave izquierda ({) y termina con la llave derecha (}) correspondiente. En medio de ellas debe haber una lista de cero o más **declaraciones** separadas por punto y coma (;).

Dentro de estas llaves puede haber cualquier carácter, teniendo en cuenta que los paréntesis (()), corchetes ([]) y llaves ({ }) deben aparecer siempre con su par correspondiente y pueden anidarse. Las comillas simples (') y dobles (") también deben ir con su par, y los caracteres encerrados por ellas son analizados como una cadena.

Declaraciones

Una **declaración** puede ser vacía o consistir en una propiedad y un valor separados por dos puntos (:). Alrededor de cada uno de estos puede haber espacio en blanco.

Varias declaraciones para un mismo selector pueden organizarse en un mismo bloque de declaraciones, separando cada declaración con punto y coma (;).

De este modo, las siguientes reglas:

```
h1 { font-weight: bold }
h1 { font-size: 12px }
h1 { line-height: 14px }
h1 { font-family: Helvetica }
h1 { font-variant: normal }
h1 { font-style: normal }
```

son equivalentes a:

```
h1 {
font-weight: bold;
font-size: 12px;
line-height: 14px;
font-family: Helvetica;
font-variant: normal;
font-style: normal
}
```

Propiedad y valor

Una propiedad es un identificador. Cualquier carácter puede aparecer en el valor. Los paréntesis (()), los corchetes ([]), las llaves ({ }), las comillas simples (') y las comillas dobles (") deben ir con su par correspondiente, y los punto y coma que no formen parte de una cadena deben ser escapados. Los paréntesis, los corchetes y las llaves pueden anidarse.

La sintaxis de los **valores** se especifica separadamente para cada propiedad, pero en todos los casos, los valores están compuestos de identificadores, cadenas, números, medidas, porcentajes, URI, colores, ángulos, tiempos y frecuencias. Las palabras claves de los valores no se deben poner entre comillas dobles o simples ("... " o ' ... '), ya que dentro de las comillas, los caracteres son tomados como una cadena de texto. Así la declaración `width: "auto"`; debe escribirse como `width: auto`; para ser un código válido.

Una aplicación de usuario debe ignorar una declaración con un nombre de propiedad no válido o un valor ilícito.

Por ejemplo, observemos las siguientes reglas de estilo:

```
h1 { color: red; font-style: 12pt }
p { color: blue; font-vendor: any; font-variant: small-caps }
em em { font-style: normal }
```

La segunda declaración del selector **h1** contiene el valor ilícito: '12pt' y la segunda declaración de **p** contiene una propiedad indefinida 'font-vendor'. Un analizador de CSS debe ignorar dichas declaraciones y reducir la hoja de estilos a:

```
h1 { color: red; }
p { color: blue; font-variant: small-caps }
em em { font-style: normal }
```

Tips de construcción

Algunos puntos a tener en cuenta al generar una regla de estilo son:

- La propiedad y el valor están rodeadas por llaves "{}" y separadas entre si por dos puntos (:).

```
body { background: white; }
```

- Si un valor está formado por varias palabras, ellas tienen que estar rodeadas por comillas dobles.

```
body { font-family: "sans serif"; }
```

- Si se quiere especificar más de una propiedad para un selector, estas deben rodearse por llaves "{}" y separarse por un punto y coma (;). La última declaración también debe estar seguida del punto y coma:

```
body { background: white; color: blue; }
```

- Para hacer el código más legible, es posible **organizar una propiedad por línea**:

```
body {
background: white;
color: blue;
}
```

- **Los selectores se pueden agrupar**, cuando varios de ellos comparten las mismas declaraciones, agrupándoles en una lista separada por comas (,). Así las reglas:

```
p {
padding: 0;
margin: 0;
}
li {
padding: 0;
margin: 0;
}
h1 {
padding: 0;
margin: 0;
}
h2 {
padding: 0;
margin: 0;
}
```

Se pueden reducir a:

```
p, li, h1, h2 {
padding: 0;
margin: 0;
}
```

Agrupamiento

Cuando varios selectores comparten las mismas declaraciones, pueden ser agrupados en una lista separada por

comas.

En este ejemplo, condensamos tres reglas con declaraciones idénticas en uno. Así,

```
h1 { font-family: sans-serif }
h2 { font-family: sans-serif }
h3 { font-family: sans-serif }
```

Es equivalente a:

```
h1, h2, h3 { font-family: sans-serif }
```

CSS ofrece también otros mecanismos "resumidos", incluyendo declaraciones múltiples y propiedades resumidas.

Extensiones Específicas de Navegador

Las palabras claves y nombres de propiedades, que comienzan con '-' o '_' son reservadas para extensiones específicas del navegador, que deben tener el siguiente formato:

```
'-' + identificador navegador + '-' + nombre propiedad
 '_' + identificador navegador + '-' + nombre propiedad
```

Por ejemplo, si el navegador ZZZ agregó una propiedad para describir el color del borde derecho de un elemento, esta se puede llamar - zzz-border-east-color.

Otros ejemplos conocidos:

- -moz-box-sizing
- -moz-border-radius
- -wap-accesskey

Algunos prefijos conocidos son: **-ms-** (Microsoft), **ms-** (Microsoft Office), **-moz-** (Mozilla), **-o-** (Opera Software), **-atsc-** (Advanced Television Standards Comité), **-wap-** (The WAP Forum).

Mayúsculas/minúsculas

Las reglas siguientes siempre sostienen:

- Las hojas de estilo CSS no distinguen mayúsculas/minúsculas (case-insensitive), excepto para las partes que no están bajo el control de CSS (los valores de los atributos XHTML **id** y **class**, los nombres de las fuentes y los URI). Recuerde que en HTML los nombres de los elementos no hacen distinción entre mayúsculas y minúsculas, pero sí lo hacen en XHTML.
- En CSS los identificadores (incluyendo los nombres de los elementos, clases e ID de los selectores) pueden contener solamente los caracteres [**A-Za-z0-9**], el guión (-) y el guión bajo (_); pero no pueden comenzar con un número. También pueden contener caracteres escapados y cualquier carácter de la ISO 10646 como código numérico, por ejemplo, el identificador "B&W?" puede ser escrito como "B\&W \?" o "B\26 W\3F". Solamente las propiedades, los valores, las unidades, las pseudo-clases, los pseudo-elementos, y las reglas-arroba pueden comenzar con un guión (-); otros identificadores (Ej. nombres, clases, o identificaciones de elemento) no pueden.
- En CSS 2.1, una barra invertida (\) indica tres tipos de escapes de caracteres:
 - Dentro de una cadena, una barra invertida seguida por una nueva línea es ignorada (es decir, la cadena se evalúa para no contener la barra invertida o la nueva línea).
 - Cancela el significado de los caracteres especiales de CSS. Cualquier carácter (excepto un número hexadecimal) se puede escapar con una barra invertida para quitar su significado especial. Por ejemplo, "\" es una secuencia que consiste en una comilla doble. Los pre-procesadores de las hojas de estilo no deben quitar estas barras invertidas de una hoja de estilo debido a que ello

puede cambiar el significado de la misma.

- El escape con barra invertida permite a los autores referirse a caracteres que no pueden ponerse fácilmente en un documento. En este caso, la barra invertida es seguida hasta por seis números hexadecimales (0..9A..F), que representan los caracteres en ISO 10646 con ese número, que no debe ser cero. Si un carácter en la gama (0-9a-fA-F) sigue el número hexadecimal, el final del número debe ser declarado. Hay dos maneras de hacer eso:

- con el espacio (u otro carácter de espacio en blanco): "\26 B" ("&B"). En este caso, las aplicaciones de usuario deben tratar un par de "CR/LF" (U+000D/U+000A) como carácter del espacio en blanco.
- proporcionando exactamente 6 dígitos hexadecimales: "\000026B" ("&B").

De hecho, estos dos métodos pueden ser combinados. Sólo un carácter de espacio en blanco es ignorado después del escape hexadecimal. Adverta que esto significa que un espacio "real" después de la secuencia de escape debe ir él mismo con escape o duplicado.

- Los escapes con barra invertida se consideran siempre parte de un identificador o de una cadena (es decir, "\7B" no es puntuación, aun cuando "{" si lo es, y "\32" está permitido al comienzo de un nombre de una clase, aunque "2" no lo está).

Comentarios

Los comentarios se utilizan para explicar el código y para facilitar la edición posterior. No tienen ninguna influencia en el procesamiento, son ignorados por los navegadores y no pueden anidarse. Los comentarios comienzan con los caracteres "/*" y terminan con los caracteres "*/".

CSS también permite los delimitadores de comentarios de SGML (<!-- y -->) en determinados lugares, pero estos no delimitan comentarios CSS. Están permitidos para que las reglas de estilo que aparecen en un documento fuente HTML (en el elemento **STYLE**) puedan ocultarse en las aplicaciones de usuario anteriores a HTML 3.2.

```
/* Esto es un comentario */
p {
  color: blue; /* Otro comentario */
  background: grey;
  font-family: arial;
}
```

LLAMADO A UN ESTILO

Tal como explicamos en el capítulo "Formas de implementar los estilos" (pág. 50) del Trabajo Final de Graduación, existen estilos en línea, incrustados y externos. Veamos como se aplica cada uno de ellos dentro de un documento XHTML, asignando el color rojo al texto de un elemento **h1** con la siguiente regla CSS:

```
h1 { color: red; }
```

Estilo en línea.

Para colocar un estilo en línea a un elemento, se utiliza el **atributo style** de dicho elemento:

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

<body>
  <h1 style="color:red">Título principal</h1>
```

```
<p>El primer párrafo de prueba que contiene un texto normal.</p>
</body>
</html>
```

🔗 [Ver ejemplo con estilo en línea \(CD > ejemplos > css > implementacion > css_enlinea.html\)](#)

Hoja de estilo incrustada.

Para poner la hoja de estilo dentro de un documento, se utiliza el **elemento STYLE** en el **HEAD** del documento:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba</title>
  <style type="text/css">
    h1 { color: red; }
  </style>
</head>

<body>
  <h1>Título principal</h1>
  <p>El primer párrafo de prueba que contiene un texto normal.</p>
</body>
</html>
```

🔗 [Ver ejemplo con hoja de estilo incrustada \(CD > ejemplos > css > implementacion > css_incrustado.html\)](#)

Hoja de estilo externa.

Para una mayor flexibilidad, se recomienda la especificación de hojas de estilo externas, que pueden cambiarse sin modificar el documento XHTML fuente y pueden compartirse entre varios documentos. Para vincular una hoja externa se utiliza el **elemento LINK**, el cuál especifica:

- el tipo de vínculo: a una hoja de estilo ("stylesheet").
- la localización de la hoja de estilo a través del atributo "href".
- el tipo de hoja de estilo que se vincula: "text/css".

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba</title>
  <link rel="stylesheet" href="principal.css" type="text/css">
</head>

<body>
  <h1>Título principal</h1>
  <p>El primer párrafo de prueba que contiene un texto normal.</p>
</body>
</html>
```

Este caso, la hoja de estilo llamada **principal.css** deberá contener la siguiente declaración:

```
h1 { color: red }
```

🔗 [Ver ejemplo con hoja de estilo externa \(CD > ejemplos > css > implementacion > css_externo.html\)](#)

Nota: Accediendo a cualquiera de los archivos de ejemplo, el resultado estético final es el mismo.

APLICACIÓN

Aplicando unas reglas CSS más al ejemplo anterior obtendríamos:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba</title>
  <style type="text/css">
    body { color: black; background: silver; }
    h1 { color: red; background: black; }
  </style>
</head>

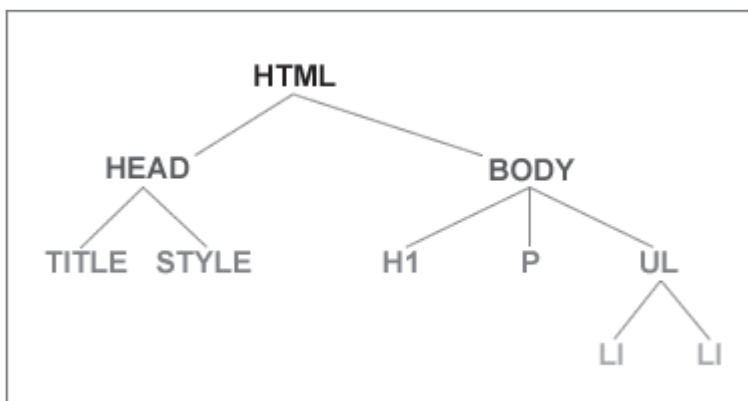
<body>
  <h1>Título principal</h1>
  <p>El primer párrafo de prueba que contiene un texto normal.</p>
  <ul>
    <li>Primer item</li>
    <li>Segundo item</li>
  </ul>
</body>
</html>
```

[Ver ejemplo de hoja de estilo incrustada con más declaraciones \(CD > ejemplos > css > implementacion > css_incrustadoplus.html\)](#)

La hoja del estilo ahora contiene cuatro declaraciones: las dos primeras establecen el color y el fondo del elemento **BODY** como `'black'` (negro) y `'silver'` (plata) respectivamente (es recomendado establecer el color del texto de fondo juntos), mientras que las dos siguientes establecen el color y el fondo del elemento **H1**, como `'red'` (rojo) y `'black'` (negro). Al no haberse especificado ningún color para el elemento **P**, este hereda el color de su elemento padre **BODY**. Si bien el elemento **H1** es también un elemento hijo de **BODY**, la tercer declaración (`color:red;`) elimina el valor heredado.

Árbol de documento

Este ejemplo da lugar al árbol siguiente:



REGLAS FRENTE A ERRORES

En algunos casos, las aplicaciones de usuario deben ignorar parte de una hoja de estilo ilícita, analizando la parte ilícita

(con el fin de localizar su comienzo y su final) pero actuando como si la misma no existiera.

Las aplicaciones de usuario cumplen con las siguientes reglas cuando se encuentran con estos casos:

PROPIEDADES DESCONOCIDAS.

Las aplicaciones de usuario deben ignorar una declaración con una propiedad desconocida. Por ejemplo, si la hoja de estilo es:

```
h1 { color: red; rotation: 70minutes }
```

La aplicación de usuario tratará esto como si la hoja de estilo hubiera sido

```
h1 { color: red; }
```

VALORES ILÍCITOS.

Las aplicaciones de usuario deben ignorar una declaración con un valor ilícito. Por ejemplo:

```
img {  
  float: left;  
  background: "red";  
  border-width: 3;  
}
```

El analizador de CSS admitirá la primera regla e ignorará background por estar su valor entre comillas, y el border por la falta de la unidad de medida:

```
img { float: left; }
```

DECLARACIONES INCORRECTAS.

Las aplicaciones de usuario deben manejar la concordancia de los pares `()`, `[]`, `{}`, `"`, `'`, y los escapes. Por ejemplo una declaración mal formulada puede omitir en la propiedad los dos puntos `:` o el valor, tal como en el siguiente ejemplo no válido, en el que sólo se coloca la propiedad color:

```
p { color:green; color }
```

PALABRAS CLAVE-ARROBA NO VÁLIDAS (ILÍCITAS).

Las aplicaciones de usuario deben ignorar una palabra clave-arroba no válida junto con todo lo que le sigue hasta el punto y coma `;` o bloque `{...}` siguiente, incluyéndolo, cualquiera que sea el que aparezca primero. Por ejemplo, considere lo siguiente:

```
@loquesea {  
  @otracoa { margin: 20px 10px; }  
  p { text-transform: uppercase; }  
}  
h1 { text-align: center; }
```

La regla-arroba `'@loquesea'` no forma parte de CSS y por consiguiente, la regla-arroba entera (hasta la tercera llave derecha incluida) es ignorada y se reduce la hoja de estilo a:

```
h1 { text-align: center; }
```

VALORES

Enteros

Un **<entero>** consiste en uno o más dígitos de cero (0) a nueve (9). Puede estar precedido por un signo más (+) o menos (-) para indicar si es positivo o negativo. Ej. 6, -8, +2. Cada propiedad establece si acepta o no valores

negativos.

Números

Un **<número>** es un valor con número reales que puede ser un **<entero>** o puede ser cero o más (entre 0 y 9) seguidos por un punto (.) y uno o más dígitos. Puede estar precedido por un signo más (+) o menos (-) para indicar si es positivo o negativo. Ej. 7, +5.6, -8.1. Cada propiedad establece si acepta o no valores negativos.

Medidas

Una **<medida>** se refiere a las dimensiones horizontales y verticales. Su formato es un signo opcional '+' (predeterminado) o '-', seguido inmediatamente por un **<número>** (con o sin punto decimal) y un identificador de unidad (ej., px, deg, etc.). Esto resultaría en: +10px o -22deg.

Cuando la medida es cero (0) el identificador de la unidad es opcional y por lo tanto es igual poner 0 y 0px.

Si bien algunas propiedades permiten valores negativos, se pueden generar limitaciones propias de la implementación. Por otra parte si un valor de medida negativo no es soportado, este debe ser convertido al valor más cercano aceptado y si se pone un valor negativo en una propiedad que no los permite, la declaración será ignorada.

Hay dos tipos de unidades de medida, las **medidas relativas** y **absolutas**.

Unidades de medida relativas

Las unidades de **medidas relativas** especifican una medida en relación a otra propiedad de medida. Las hojas de estilo que las utilizan permiten una adaptación más fácil de su escala de un medio a otro (ej., de un monitor a una impresora).

Tal como lo indica la W3C⁽¹⁾, las unidades relativas son:

- **em**: el tamaño ('font-size') de la fuente relevante
- **ex**: la 'altura de la x' de la fuente relevante
- **px**: píxeles, relacionado con los dispositivos visuales

Cuando **em** y **ex** son especificadas en la raíz de la estructura del documento (Ej. **HTML**), estas se refieren al valor inicial de la propiedad.

Los elementos hijos no heredan los valores relativos especificados para sus padres; ellos (generalmente) heredan los valores computados.

[🔗 Ver ejemplo de medidas relativas \(CD > ejemplos > css > valores > medidas_relativas.html\)](#)

Unidades de medida absoluta

Las **unidades de medida** absoluta son útiles solamente cuando se conocen las propiedades físicas del medio de salida. Las unidades absolutas son

- **in**: inches — 1 pulgada (inch) es igual a 2.54 centímetros.
- **cm**: centímetros
- **mm**: milímetros
- **pt**: puntos — los puntos equivalen a 1/72 pulgadas.
- **pc**: picas — 1 pica es igual a 12 puntos.

1 <http://www.w3.org/TR/CSS21/syndata.html#length-units>

En caso de que la medida especificada no pueda ser soportada, las aplicaciones de usuario deben aproximarla al valor real.

Ver ejemplo de medidas absolutas (CD > ejemplos > css > valores > medidas_absolutas.html)

Porcentajes

Un **<porcentaje>** es un **<número>** seguido inmediatamente por '%'. Puede estar precedido por los signos de más (+) o menos (-) para indicar si es positivo o negativo. Los valores expresados en porcentajes son siempre relativos a otro valor.

Las propiedades que admiten porcentajes definen el valor al cual se refiere el porcentaje (otra propiedad para el mismo elemento, una propiedad para un elemento antepasado o un valor en el contexto del formato)

Quando un porcentaje es atribuido a una propiedad del elemento raíz y el porcentaje es definido en referencia al valor heredado de alguna propiedad, el valor resultante es la cantidad del porcentaje del valor inicial de esta propiedad.

En el siguiente ejemplo, el alto de la línea es **120%** mayor que el tamaño de la fuente, y el **strong** es un **110%** mayor que la fuente del **p**:

```
p {
  font-size: 10px;
  line-height: 120% }
strong { font-size: 110% }
```

lo que daría como resultado:

Lorenlī molor ipīs adignīb ex ex ecte comy **nulla feummodit** luptat. Duis nos nulla eugue vulputat, quisit aut ut praesse faccum zzriureet ulput laore conulla faciliquisi.

URI

Los URI (Identificadores Uniformes de Recursos) denotados como `<uri>` proporcionan la dirección de un recurso en la Web y en el valor de una propiedad CSS los URI se designan como `url()`. Dentro de los paréntesis va el URI propiamente dicho, entre comillas dobles o simples opcionales. Por lo tanto, una URI puede tomar cualquiera de los siguientes formatos:

Con comillas dobles:

```
body { background: url("http://www.dominio.com/imagen.gif") }
```

Con comillas simples:

```
body { background: url('http://www.dominio.com/imagen.gif') }
```

Sin comillas:

```
body { background: url(http://www.dominio.com/imagen.gif) }
```

Los paréntesis, las comas, los espacios en blanco, las comillas simples (') y las comillas dobles (") que aparecen en un URI deben ir escapadas con una barra invertida: \ ' (,) \ ' \ " .

Se pueden utilizar URIs absolutos como los anteriores, que indican la dirección completa o URIs relativos que son convertidos a URIs completos usando un URI de base. Para las hojas de estilo CSS, los URI relativos lo son con respecto al archivo que contiene la hoja de estilo.

Por ejemplo, si la siguiente regla está localizada en el archivo designado por la URI: <http://www.dominio.com/>

estilos/basic.css

```
body { background: url("cuadros") }
```

El fondo de **BODY** en el documento fuente será la imagen `cuadros` que se encuentra dentro de la misma carpeta "estilos", exactamente en la siguiente URL: `http://www.dominio.com/estilos/cuadros`

Contadores

Para referirse al valor de un contador se usa la notación:

- `counter(<identificador>)`
- `counter(<identificador>, <estilo-de-lista>)` siendo "decimal" el estilo predeterminado.

Para referirse a una secuencia de contadores anidados la notación es:

- `counters(<identificador>, <cadena>)`
- `counters(<identificador>, <cadena>, <estilo-de-lista>)`

Ver "Contadores anidados y área de alcance" en el capítulo sobre contenido generado.[]

El valor de los contadores sólo puede ser referenciado desde la propiedad '`content`' y '`none`' es un posible <estilo-de-lista>: '`counter(x, none)`' genera una cadena vacía.

Aquí hay una hoja de estilo que numera los párrafos (**P**) para cada capítulo (**H1**). Los párrafos son numerados con números romanos, seguidos por un punto y un espacio:

```
p {counter-increment: par-num}
h1 {counter-reset: par-num}
p:before {content: counter(par-num, upper-roman) ". "}
```

Colores

Un `<color>` puede ser especificado por una palabra clave o por una especificación numérica.

Palabras claves: `aqua`, `black`, `blue`, `fuchsia`, `gray`, `green`, `lime`, `maroon`, `navy`, `olive`, `orange`, `purple`, `red`, `silver`, `teal`, `white` y `yellow`.

Especificación numérica: Se utiliza el modelo de color RGB (Red, Green, Blue que es igual a rojo, verde y azul).

- **RGB hexadecimal:** signo numeral (#) seguido inmediatamente por 6 caracteres hexadecimales (0123456789ABCDEF), siendo el resultado `#rrggbb`. Ej.: el color rojo sería `#FF0000`.
- **RGB hexadecimal abreviado:** se utiliza cuando se repiten los dígitos siendo el formato `#rgb`. Ej.: el color rojo sería `#F00`.
- **rgb(r,g,b):** notación funcional de RGB que permite especificar 256 niveles de luminosidad para cada uno de los colores (rojo, verde o azul). Tiene el formato `rgb()` y dentro de los paréntesis va una lista de tres valores numéricos separados por comas. El cero (0) significa la ausencia del color y 255 el color puro. Así, el negro (Rojo 0, Verde 0, Azul 0) está dado por la ausencia de los tres colores primarios y el blanco por la suma de ellos (Rojo 255, Verde 255, Azul 255). Son 256 colores iniciando por cero (0), por ello el número más alto es el 255.
- **rgb(r%,g%,b%):** Es igual que la notación anterior, sólo que se utilizan porcentajes en lugar de valores enteros, y el 255 está representado por el 100%.

Todos estos ejemplos especifican el mismo color en cada una de las formas mencionadas arriba:

```
p { color: #f00 } /*#rgb*/
p { color: #ff0000 } /*#rrggbb*/
p { color: rgb(255,0,0) } /*rango de enteros 0 - 255*/
p { color: rgb(100%, 0%, 0%) } /*rango de porcentajes 0.0% - 100.0%*/
```

Entonces, el valor entero 255 corresponde a 100%, y a F o FF en la notación hexadecimal. En base a esto: `rgb(255, 255, 255) = rgb(100%, 100%, 100%) = #FFF`.

🔗 [Ver tabla de colores según palabras claves \(CD > ejemplos > css > valores > colores.html#colores_claves\)](#)

Al indicar los colores en cualquiera de los métodos señalados con anterioridad surge un inconveniente de compatibilidad entre la paleta de Windows y la de Mac. Ambos sistemas tienen algunos colores en común, que conforman lo que se conoce como **paleta Web o paleta segura** (safety palette). Esta paleta segura está basada en los valores RGB con incrementos de 20% entre cada valor, lo que da como resultado 216 colores.

Si bien las aplicaciones de usuario pueden variar en la fidelidad con la que representan estos colores, el uso de sRGB proporciona seguridad en cuanto a su presencia en cualquier sistema.

Los valores de rojo, verde y azul a veces tienen que ser cambiados para adecuarse al rango soportado por el dispositivo. Si para un monitor CRT típico, cuya gama es la misma que sRGB, se indica el color: `rgb(300, 0, 0)` se deberá recortar el color y dejarlo en el valor más cercano a 300, que en este caso es el 255: `rgb(255, 0, 0)`

En otros dispositivos como las impresoras, que tiene gamas distintas a sRGB, algunos colores fuera del rango (0-255) de sRGB serán representables, mientras que colores dentro del rango estarán fuera de la gama del dispositivo y de ese modo serán recortados.

🔗 [Ver tabla de colores SRGB \(CD > ejemplos > css > valores > colores.html#colores_srgb #808080\)](#)

Además de estas palabras clave y especificaciones numéricas, autores y usuarios pueden utilizar palabras clave que corresponden a los colores usados por ciertos objetos en el entorno del usuario. (Ver "[Colores Del Sistema CSS2](#)")

🔗 [Ver tabla de colores del sistema \(CD > ejemplos > css > valores > colores.html#colores_sistema\)](#)

Nota sobre el uso de colores: Aunque los colores pueden añadir información a los documentos y hacerlos más legibles, no se deben utilizar combinaciones de colores que causen problemas a personas con dificultades para distinguir colores (daltónicos, con monitores monocromo, etc.). Al establecer un fondo o imagen de fondo (`'background'`), se recomienda establecer también el color de los textos (`'color'`).

Cadenas

Las cadenas pueden escribirse con comillas dobles o simples. Dentro de una comilla doble sólo puede aparecer una comilla simple y viceversa. Las comillas dobles no pueden aparecer dentro de otras comillas dobles a menos que sean escapadas (como `\` o como `\22`) y lo mismo ocurre con las comillas simples (`'\'` o `'\27'`):

```
"una 'cadena'"
"una \"cadena\""
'una "cadena"'
'una \'cadena\''
```

Dos ejemplos correctos de cadenas son:

```
p {font-family: "Times New Roman"}
p {font-family: 'Times New Roman'}
```

Las palabras clave no deben ir entre comillas pues se convierten en cadenas: Ej.: `red` es una palabra clave, `"red"` es una cadena de texto. Una cadena vacía se indica (`""`).

Una cadena no puede contener una nueva línea directamente, y para incluir una nueva línea se debe usar un escape representado por el carácter (U+000A), `"\A"` o `"\0000a"`.

Identificadores

Los identificadores (incluyendo los nombres de los elementos, clases e ID de los selectores) pueden contener sólo caracteres y el guión (-); pero no pueden comenzar con un guión o un número.

Forma

El único valor permitido para la forma es: `rect(top, right, bottom, left)`. **Arriba, derecha, abajo, izquierda** especifican los desplazamientos de los lados respectivos de la caja y pueden tener un valor de `<medida>` o "auto" (significa lo mismo que "0"). Las medidas negativas están permitidas. Ejemplo:

```
P { clip: rect(5px, 10px, 10px, 5px) }
```

Inherit

"Inherit" (o heredado) significa que el elemento toma el mismo valor computado que la propiedad del padre del elemento. De este modo, se refuerza explícitamente el valor heredado. Pero, además, este valor se aplica aún a propiedades que de otro modo no serían heredadas.

Valores no soportados

Si una AU (Aplicación de usuario) no soporta un valor en particular, esta debe ignorar ese valor como si ese valor fuera un valor ilícito. Por ejemplo:

```
h3 {  
  display: inline;  
  display: run-in;  
}
```

Una AU que soporta el valor `'run-in'` para la propiedad `'display'`, aceptará primero el valor del primer `'display'`, el cual será sobre escrito por el valor de la segunda declaración de `'display'`.

Una AU (Aplicación de usuario) que no soporte el valor `'run-in'` procesará el primer valor de la declaración del `display` e ignorará el valor de la segunda declaración del `display`.

CODIFICACIÓN Y CARACTERES

Una hoja de estilo es una secuencia de caracteres del Conjunto Universal de Caracteres, que deben ser codificados, para su transmisión y almacenamiento, por una codificación de caracteres que soporte el conjunto de caracteres disponibles en US-ASCII (ej., ISO 8859-x, SHIFT JIS, etc.).

Cuando una hoja de estilo está incrustada en otro documento, tal como en el elemento **STYLE** o el atributo **style** de XHTML, la hoja de estilo comparte la misma codificación de caracteres que todo el documento. Sin embargo, cuando una hoja de estilo se encuentra en un archivo separado, las aplicaciones de usuario deben observar las siguientes prioridades cuando determinan la codificación de caracteres del documento (de la prioridad más alta a la más baja):

1. Un parámetro HTTP "charset" en un campo "Content-Type" (o parámetro similar en otro protocolo)
2. BOM y/o `@charset`
3. `<link charset="">` u otro metadata del mecanismo de vinculación
4. Mecanismos de referencia de las hojas de estilos o del documento
5. Asumir UTF-8

Una regla `@charset` puede aparecer al comienzo de una hoja de estilo externa, sin ser precedida por ningún carácter, y seguida del nombre de una codificación de caracteres:

```
@charset "ISO-8859-1";
```

3. Selectores



EQUIVALENCIA DE PATRONES

Las reglas de equivalencias de patrones determinan qué reglas del estilo se aplican a los elementos en la estructura del documento*. Estos patrones, llamados **selectores**, van desde el nombre de un elemento hasta complejos patrones contextuales. Si todas las condiciones en el patrón son verdaderas para un elemento, entonces el selector equivale al elemento.

La distinción entre mayúsculas/minúsculas (case-sensitivity) en los nombres de los elementos de los selectores depende del lenguaje del documento. Por ejemplo, en HTML, los nombres son insensibles a la diferencia entre mayúsculas/minúsculas (case-insensitive), pero en XHTML sí hacen distinción entre ellas (case-sensitive).

La siguiente tabla resume la sintaxis de los selectores:

Patrón	Significado	Descrito en:
*	Equivale a cualquier elemento.	Selector Universal
E	Equivale a cualquier elemento E (ej., un elemento del tipo E).	Selectores de tipo
E F	Equivale a cualquier elemento F que sea un descendiente de un elemento E.	Selectores de descendientes
E > F	Equivale a cualquier elemento F que sea hijo de un elemento E.	Selectores de hijos
E:first-child	Equivale al elemento E cuando E es el primer hijo de su padre.	La pseudo-clase de:first-child
E:link E:visited	Equivale al elemento E si E es el ancla de origen de un hipervínculo cuyo destino aún no se ha visitado (:link) o ya se visitó (:visited).	Las pseudo-clases :link

Patrón	Significado	Descrito en:
E:active E:hover E:focus	Equivale a E durante ciertas acciones del usuario.	Las pseudo-clases dinámicas
E:lang(c)	Equivale a un elemento de tipo E si está en el lenguaje (humano) c (el lenguaje del documento especifica cómo se determina el idioma).	la pseudo-clase :lang()
E + F	Equivale a cualquier elemento F precedido inmediatamente por un elemento E hermano.	Selectores adyacentes
E[foo]	Equivale a cualquier elemento E con el atributo "foo" (cualquier valor).	Selectores de atributo
E[foo="warning"]	Equivale a cualquier elemento E cuyo valor del atributo "foo" sea exactamente igual a "warning".	
E[foo~="warning"]	Equivale a cualquier elemento E cuyo atributo "foo" tiene un valor consistente en una lista de valores separados por espacios, uno de los cuales es exactamente igual a "warning".	
E[lang]="en"]	Equivale a cualquier elemento E cuyo atributo "lang" tiene una lista de valores separados por guiones que comienzan (desde la izquierda) con "en".	Selectores de clase
DIV.warning	Específico del lenguaje. (en HTML, igual que DIV[class~="warning"].)	
E#myid	Equivale a cualquier elemento E cuyo ID es igual a "myid".	Selectores de ID

SINTAXIS DE LOS SELECTORES

Un **selector simple** es un [selector de tipo](#) o [selector universal](#) seguido inmediatamente por cero o más [selectores de atributos](#), [selectores de ID](#), o [pseudo-clases](#), en cualquier orden. El selector simple es equivalente si todos sus componentes son equivalentes, lo que significa que las propiedades declaradas por dicha regla sólo serán aplicadas en la estructura del documento a los elementos correspondientes sólo si todos los componentes del selector son equivalentes en la estructura del documento con los elementos (y atributos) a los cuáles hacen referencia.

Un selector es una cadena de uno o más selectores simples separados por combinadores: espacio en blanco, ">", y "+". El espacio en blanco puede aparecer entre un combinador y los selectores simples que están próximos a él.

Los elementos de la estructura del documento que equivalen a un selector son llamados **sujetos del selector**. Así, un selector que consiste en un solo selector simple se corresponde con cualquier elemento que satisface sus requisitos. Sin embargo, al anteponer un selector simple y un combinador a una cadena se imponen restricciones para la equivalencia, con lo que los sujetos de un selector son un sub-conjunto de los elementos que equivalen al último selector simple.

Un [pseudo-elemento](#) se puede añadir al selector simple pasado en una cadena, en cuyo caso la información del estilo se aplica a una sub-parte de cada sujeto

Selector universal

Formato: *

El **selector universal** (*) es equivalente con cualquier elemento en la estructura del documento. Si el selector universal no es el único componente de un [selector simple](#), el "*" puede ser omitido. Por ejemplo: *.myclass y .myclass son equivalentes, así como también lo son *#myid y #myid.

Selectores de tipos

Formato: *E*

Un **selector de tipo** se corresponde con el nombre de un tipo de elemento en el lenguaje del documento y equivale con cada instancia de ese tipo de elemento en la estructura del documento.

Estos selectores tienen una ventaja que a la vez se convierte en su limitación: permiten modificar el aspecto de todos los elementos de un tipo (por ejemplo, **H1**) en todas las páginas del sitio, pero a la vez limita a todos esos elementos de un tipo a la misma apariencia en todas las páginas.

La regla siguiente engloba a todos los elementos **H1** de la estructura del documento:

```
h1 { font-family: sans-serif; }
```

Selectores de descendiente

Formato: *E F*

Los **selectores de descendientes** se corresponden con un elemento que es el descendiente de otro elemento en la estructura del documento (ej., todos los elementos **STRONG** contenidos por un **H2**). Se componen de dos o más selectores separados por un espacio en blanco. Un selector de descendiente de la forma "**E F**" engloba a **F** cuando es descendiente de algún antepasado* de **E**.

Ejemplos

Ejemplo 1

CÓDIGO

Si en nuestro CSS tenemos las reglas que vemos a continuación:

```
h2 { color: blue; }  
strong { color: blue; }
```

En el caso que un elemento **STRONG** aparezca dentro de un elemento **H2**, la distinción en azul del elemento **STRONG** sería irrelevante por su presencia dentro de un elemento **H2** cuyo texto también se encuentra en color azul.

```
<h2>Un título con <strong>negrita</strong> en su contenido.</h2>  
<p>Un texto con un <strong>strong</strong> en cualquier otro lugar.</p>
```

Si queremos resaltar el elemento **STRONG** ubicado dentro del elemento **H2**, entonces debemos crear una tercera regla como se muestra a continuación:

```
h2 { color: blue; }  
strong { color: blue; }  
h2 strong { color: black; }
```

Esta regla indicará que cualquier elemento **STRONG** que se encuentre dentro de un elemento **H2** deberá mostrarse en color negro y no en color azul como lo indica la segunda regla. El resultado sería el siguiente:

RESULTADO

Un título con negrita en su contenido.

Un texto con un **strong** en cualquier otro lugar.

Ejemplo 2

CÓDIGO

```
div * p
```

El uso del `*` empareja un elemento de **P** descendiente más lejano de un elemento **DIV**. El uso opcional del espacio en blanco a ambos lados del `"*"` que no forma parte del selector universal; el espacio en blanco es un combinador que indica que el **DIV** debe ser el antepasado de un cierto elemento, y que ese elemento debe ser un antepasado de **P**.

Ejemplo 3

CÓDIGO

```
div p * [ href ]
```

Este selector combina los selectores descendientes con los selectores de atributos, empareja cualquier elemento que tiene asignado el atributo **href** y que se encuentra dentro de un **P** que a su vez está dentro de un **DIV**.

Selectores hijos

Formato: $E > F$

Un **selector hijo** engloba a un elemento que es hijo* de un cierto elemento y se compone de dos o más selectores separados por `">"`.

CÓDIGO 1

La regla siguiente fija el estilo de todos los elementos de **LI** que sean hijos de **BODY**:

```
BODY > li { line-height: 1.3; }
```

CÓDIGO 2

Y este ejemplo combina los selectores descendientes y los selectores hijo:

```
DIV OL>LI P { color: red; }
```

Equivale a un elemento **P** que sea un descendiente de un **LI**; el elemento **LI** debe ser el hijo de un elemento **OL**; el elemento **OL** debe ser un descendiente de un **DIV**.

Selectores de hermanos adyacentes

Formato: $E1 + E2$

En los selectores de **hermanos adyacentes** **E2** es el sujeto del selector. El selector equivale si **E1** y **E2** comparten el mismo padre en la estructura del documento y **E1** precede inmediatamente a **E2**, ignorando los nodos no-elementos (tales como nodos de texto y comentarios).

CÓDIGO 1

Así en el ejemplo siguiente si un **H2** se encuentra inmediatamente después de un **H1**, se reduce el espacio vertical que separa al **H2** del **H1**:

```
h1 + h2 { margin-top: -5mm }
```

CÓDIGO 2

Esta regla se puede aplicar también sólo cuando los **H1** tienen especificada una clase (`class="prueba"`) en cuyo caso, el código sería el siguiente:

```
h1.prueba + h2 { margin-top: -5mm }
```

Selectores de atributos

Los **selectores de atributos** sirven para especificar reglas que sean equivalentes con atributos de un tipo de elemento definido en el documento fuente. Por ejemplo, especificar una fuente especial para aquellos títulos que tienen asignado el atributo **title**.

Los selectores de atributos pueden equivaler de cuatro maneras, y siempre hay que tener en cuenta que los valores del atributo deben ser identificadores o cadenas (la distinción entre mayúsculas/minúsculas en los nombres y valores de atributos depende del lenguaje del documento):

🔗 [Ver ejemplo de selectores de atributos \(CD > ejemplos > css > selectores > selectores_atributos.html\)](#)

E[att]

Equivale a cualquier elemento **E** que tiene asignado el atributo "**att**", cualquiera que sea el valor del atributo.

CÓDIGO CSS

Podemos querer asignar un estilo especial (**color:red;**) a todos los párrafos (**P**) que tengan declarado el atributo **title** cualquiera sea su valor:

```
p[title] { color: red; }
```

CÓDIGO XHTML

```
<p>Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat. Duis nos  
nulla eugue vulputat, quisit aut ut praesse faccum zzriureet ulput laore conulla  
faciliquisi. </p>  
<p title="Los versos de Bach">Lorpercipit nulla faci tat prat, sit prat. Dui et  
aliquat adiam, quissequat nibh eugait veniamcommy nons nibh essi. </p>
```

RESULTADO

En la siguiente imagen podemos visualizar el resultado. El primer párrafo es mostrado con su estilo normal, pero como en el segundo se produce una equivalencia, por tener asignado un atributo **title** con cualquier valor, el mismo se muestra en color rojo:

Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat.
Duis nos nulla eugue vulputat, quisit aut ut praesse faccum zzriureet
ulput laore conulla faciliquisi.

Lorpercipit nulla faci tat prat, sit prat. Dui et aliquat adiam, quissequat
nibh eugait veniamcommy nons nibh essi.

E[att=val]

Equivale a cualquier elemento **E** que tiene asignado el atributo "**att**", cuyo valor es exactamente igual a "**val**".

CÓDIGO CSS

Se puede utilizar para asignar un estilo especial a todos los títulos **H1** que tienen el atributo **class** con el valor **importante**:

```
h1[class=importante] { text-transform: uppercase; }
```

CÓDIGO XHTML


```
<h1>Primer título normal</h1>
<p>Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat. Duis nos
nulla eugue vulputat, quisit aut ut praesse faccum zzriureet ulput laore conulla
faciliquisi.</p>
...
<h1 class="importante">Segundo título con estilo</p>
<p>Lorpercipit nulla faci tat prat, sit prat. Dui et aliquat adiam, quissequat
nibh eugait veniamcommy nons nibh essi.</p>
```

RESULTADO

Lo que daría como resultado, la siguiente imagen, donde el segundo encabezado de nivel 1 es mostrado en mayúsculas y azul por encontrarse una equivalencia en el atributo **class** con el valor exacto **importante**:



E[att~=val]

Equivale a cualquier elemento **E**, cuando el valor del atributo "att" del elemento es una lista de palabras separadas por espacios, y una de las cuales es exactamente igual a "val". Si se utiliza este selector, las palabras en el valor no deben contener espacios (puesto que son separadas por espacios). Con el teclado puede incorporar en el texto el símbolo alterno (~) presionando las teclas **ALT + 126**.

CÓDIGO CSS

Este selector se puede utilizar para asignar un estilo especial a todos los párrafos que tienen en su atributo **title**, la palabra "versos" entre otras palabras separadas por espacios:

```
p[title~=versos] { font-style: italic; }
```

CÓDIGO XHTML

```
<p title="Los versos de Rowling">Duipit la facilluptat, consent dolesequi eumsan
ulluptat lorperiure magnis del utat.</p>
<p>Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat. Duis nos
nulla eugue vulputat, quisit aut ut praesse faccum zzriureet ulput laore conulla
faciliquisi.</p>
<p title="Los versos de Bach">Lorpercipit nulla faci tat prat, sit prat. Dui et
aliquat adiam, quissequat nibh eugait veniamcommy nons nibh essi.</p>
```

RESULTADO

En esta imagen los dos párrafos (el primero y tercero) que tienen dentro de **title** varias palabras separadas por espacios, y una de ellas es "versos" son mostradas en rojo e itálica:

Duipit la facilluptat, consent dolesequi eumsan ulluptat lorperiure magnis del utat.

Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat. Duis nos nulla eugue vulputat, quisit aut ut praesse faccum zzriureet ulput laore conulla faciliquisi.

Lorpercipit nulla faci tat prat, sit prat. Dui et aliquat adiam, quissequat nibh eugait veniamcommy nons nibh essi.

E[att|=val]

Equivale a cualquier elemento **E**, cuando el valor del atributo "**att**" del elemento es una lista de "palabras" separadas por guiones, comenzando con "**val**". La correspondencia comienza al principio del valor del atributo y está pensado principalmente para permitir equivalencias con el sub-código del lenguaje (Ej. el atributo **lang** en XHTML).

CÓDIGO CSS

La siguiente regla será equivalente a valores del atributo "**lang**" que empiecen con "**en**", incluyendo "**en**", "**en-US**" y "**en-cockney**":

```
*[lang|="en"] { color : blue; }
```

CÓDIGO XHTML

```
<p lang="en-US">Duipit la facilluptat, consent dolesequi eumsan ulluptat
lorperiure magnis del utat.</p>
<p>Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat. Duis nos
nulla eugue vulputat, quisit aut ut praesse faccum zzriureet ulput laore conulla
faciliquisi.</p>
<p lang="en-cockney">Lorpercipit nulla faci tat prat, sit prat. Dui et aliquat
adiam, quissequat nibh eugait veniamcommy nons nibh essi.</p>
```

RESULTADO

Esto buscaría cualquier elemento (*) donde el valor de **lang** comience con en para mostrarlos en un texto azul:

Duipit la facilluptat, consent dolesequi eumsan ulluptat lorperiure magnis del utat.

Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat. Duis nos nulla eugue vulputat, quisit aut ut praesse faccum zzriureet ulput laore conulla faciliquisi.

Lorpercipit nulla faci tat prat, sit prat. Dui et aliquat adiam, quissequat nibh eugait veniamcommy nons nibh essi.

Combinaciones de selectores de atributos

CÓDIGO CSS

Se pueden utilizar combinaciones de los cuatro tipos de selectores de atributos mencionados arriba, para asignar por ejemplo un estilo de letra gris y mayúscula a aquellos elementos **EM** que son descendientes de un **P** y que tienen un atributo **class="importante"** y un **title** cualquiera sea su valor:

```
p em[class="importante"][title] { color: grey; text-transform: uppercase; }
```

CÓDIGO XHTML

```
<p>Lorerili molor ipis <em>adignibh</em> ex ex ecte commy nulla feummodit luptat.
<em title="La clave de la vida">Duis nos nulla</em> eugue vulputat, quisit aut ut
praesse faccum <em class="importante" title="Nuestra salvación">zzriureet ulput
laore</em> conulla faciliquisi.</p>
```

RESULTADO

Como el primer **EM** no tiene asignado ni un **title** ni un **class="importante"** y el segundo **EM** sólo tiene el **"title"** la regla no se aplica a ellos. Sin embargo, el tercer **EM** es descendiente de un párrafo y aúna un **"title"** con cualquier valor y el atributo **"class"** con el valor **"importante"** por lo que la regla tiene equivalencia y el mismo debe ser mostrado en azul y mayúsculas:

```
Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat.
Duis nos nulla eugue vulputat, quisit aut ut praesse faccum ZZRIUREET
ULPUT LAORE conulla faciliquisi.
```

Selectores de clases

Como ya indicamos, los selectores de tipo tienen la ventaja y limitación de modificar el aspecto de todos los elementos de un tipo (por ejemplo, **P**) en todas las páginas del sitio. Cuando queremos que un párrafo tenga características diferenciadas del resto de los párrafos en la estructura del documento, debemos crear una clase de párrafo.

Entonces, con un selector de clases se pueden definir diferentes estilos para un mismo tipo de elemento HTML. En la hoja de estilos CSS, el selector de clases consta de un punto (.) seguido inmediatamente por el nombre de la clase inventada por el autor:

```
p.miclase { color: green }
```

En la etiqueta de la página HTML/XHTML se coloca el atributo **class** seguido de un signo igual (=) y del nombre de la clase entre comillas (""). En el documento XHTML se pueden colocar cualquier cantidad de llamados a una misma clase como sean necesarios.

```
<p class="miclase">Lor in vulputat. Lor sed ex et nit augait, sisi.</p>
<p>Lore mincing euguer aliquat ilit illaor alisl delenibh eu feuis dit wisissenis
nullutat. Duip erat dip ea core feuisi. </p>
```

Según el código anterior, el color verde será aplicado al primer párrafo por tener el atributo **class="miclase"** pero no así al segundo párrafo.

Si bien estos selectores requieren agregar el atributo **class** en el elemento HTML, permiten alterar la presentación de elementos específicos. También admiten la creación de una clase genérica para aplicar un determinado estilo a cualquier elemento de la página al que la clase esté asignada. Para esto sólo hay que dejar únicamente el punto (.) y el nombre de la clase:

```
.miclase { color: green }
```

Con esta regla se asignará un color verde al texto de todos los elementos a los cuáles les aplicó la clase **"miclase"**:

```
<h1 class="miclase">Un título de nivel 1 con miclase.</h1>
<p class="miclase">Lor in vulputat. Lor sed ex et nit augait, sisi.</p>
```

Para aplicar más de una clase a un elemento dado, la sintaxis es:

```
<p class="miclase otro">Lor in vulputat. Lor sed ex et nit augait, sisi.</p>
```

Al párrafo anterior se lo mostrará con la clase **"miclase"** y la clase **"otro"**.

Para equivaler a un subconjunto de valores **class**, cada valor debe ir precedido por un ".", en cualquier orden.

Por ejemplo, la siguiente regla equivale a cualquier elemento **P** cuyo atributo **class** le ha sido asignado una lista de valores separados por espacios que incluyen "aprobado" y "marino":

```
p.aprobado.marino { color: green; }
```

Esta regla equivale en el caso de `class="aprobado rojo azul marino"` pero no se corresponde en `class="aprobado rojo"`.

Nota: "DIV.valor" y "DIV[class~=valor]" tiene el mismo significado.

Importante: Un nombre de clase no debe comenzar con un número porque no funcionará en Mozilla/Firefox.

Selectores de ID

Los **selectores de ID** permiten aplicar estilos a un sólo elemento de la página de manera excluyente y se identifican con el símbolo numeral (#) inmediatamente seguido por el valor de **id**. Si un elemento tiene asignado el atributo `id="importante"` no podrá haber otro elemento con el mismo valor asignado. Los selectores de ID no tienen la flexibilidad de los selectores de clases, pero son útiles para identificar de modo exclusivo un determinado elemento de la página.

```
h1#titular { text-align: center; }  
#bajada { font-style: italic; font-weight: bold; }
```

Con las reglas anteriores, los elementos **H1** identificados con el **id** "titular" serán centrados, y cualquier elemento al que se le ha asignado el **id** "bajada" irá en negritas e itálicas.

En el siguiente ejemplo, la regla de estilo equivale al elemento que tiene el valor de ID "importante", que en este caso es el elemento **P**:

```
<head>  
<title>Documento de prueba</title>  
<style type="text/css">  
*#importante { font-weight: bold; color: red; }  
</style>  
</head>  
<body>  
<p id="importante">Texto importante en negrita rojo.</p>  
</body>
```

Si reemplazáramos `*#importante`, por `h1#importante` la regla de estilo sólo se aplicaría a una encabezado de nivel 1 que tuviera el **id** "importante".

Los selectores de ID tienen una precedencia mayor que los selectores de atributos. Por ejemplo, en HTML, el selector `#p123` es más específico que `[ID=p123]` en términos de cascada.

Importante: Un nombre de **id** no debe comenzar con un número porque no funcionará en Mozilla/Firefox.

Pseudo-clases

Las **pseudo-clases** permiten aplicar formato a información o elementos que no se deducen de la estructura del documento. Son abstracciones que permiten referirse a elementos inaccesibles y los clasifican basándose en propiedades más allá de su nombre, atributos o contenidos. Las pseudo-clases pueden ser dinámicas, ya que un elemento puede adquirir o perder una pseudo-clase a medida que el usuario interactúa con el documento. La excepción es `:first-child`, que puede deducirse de la estructura del documento, y `:lang()`, que puede deducirse de la estructura del documento en algunos casos.

Las pseudo-clases no aparecen en el documento fuente o en la estructura del documento, sin embargo son permitidas

en cualquier lugar dentro de un selector. Los nombres no hacen distinción entre mayúsculas y minúsculas (case-insensitive).

Algunas pseudo-clases son mutuamente excluyentes, mientras otras pueden aplicarse simultáneamente al mismo elemento. En caso de reglas conflictivas, el orden de cascada normal determina el resultado.

Las pseudos-clases existentes son:

- `:first-child`
- `:link` y `:visited`
- `:hover`, `:active` y `:focus`
- `:lang`

Pseudo-clase `:first-child`

La **pseudo-clase `:first-child`** equivale a un elemento que es el primer hijo de cualquier otro elemento.

Hay que tener en cuenta que cuando se quiere aplicar un estilo al primer hijo de cualquier elemento en la estructura del documento, es lo mismo colocar `* > p:first-child` y `p:first-child`. No se permiten espacios entre el nombre del elemento y la pseudos-clase, pero sí alrededor del signo mayor (<) que separa al elemento padre de su hijo.

Ejemplos

Aplicación 1:

CÓDIGO CSS

Para darle sangría al párrafo que sea primer hijo de un **DIV** debemos usar el código siguiente:

```
div > p:first-child { font-weight: bold; }
```

CÓDIGO XHTML

El selector anterior sólo sería equivalente con el primer párrafo que se encuentra dentro del **DIV** en el siguiente código HTML, el cuál sería mostrado en negrita:

```
<div class="nota">
<p>El primer párrafo dentro de la nota.</p>
<p>El segundo párrafo dentro de la nota.</p>
</div>
```

RESULTADO

El primer párrafo dentro de la nota.
El segundo párrafo dentro de la nota.

Sin embargo, si el código fuera como el que sigue, la regla no tendría equivalencias porque el párrafo no es el primer hijo del **DIV**.

```
<div class="nota">
  <h1>Titular</h1>
<p>El primer párrafo dentro de la nota.</p>
<p>El segundo párrafo dentro de la nota.</p>
</div>
```

Aplicación 2:

CÓDIGO CSS

Con el código que sigue a este párrafo, podríamos asignar el estilo de fuente en negrita para los elementos **EM** que son descendientes de un elemento **P** que es el primer hijo de un **DIV**:

```
div > p:first-child em { color : red; }
```

CÓDIGO XHTML

Utilizando los mismos códigos anteriores, podríamos agregar un elemento **EM** a cada párrafo.

```
<div class="nota">
<p>El primer párrafo <em>dentro de la nota</em>.</p>
<p>El segundo párrafo <em>dentro de la nota</em>.</p>
</div>
```

RESULTADO

Tal como lo muestra la imagen, si bien ambos párrafos tienen **EM**, sólo se aplica la regla al **EM** del primer párrafo porque es el primer hijo de **DIV**.

El primer párrafo *dentro de la nota.*
El segundo párrafo *dentro de la nota.*

🔗 Ver ejemplo de pseudo-clase :first-child (CD > ejemplos > css > selectores > pseudo_clases_firstchild.html)

Nota: Las cajas anónimas no forman parte de la estructura del documento y no se cuentan en el cálculo del primer hijo.

Pseudo-clases :link y :visited

Las aplicaciones de usuario generalmente muestran de forma diferente los vínculos no visitados y aquellos que han sido visitados con anterioridad. También, después de cierto tiempo pueden dejar de mostrar un vínculo visitado de forma diferente y volverlo a su condición de no-visitado.

Gracias a CSS y sus **pseudos-clases :link y :visited** (mutuamente excluyentes) se puede hacer dicha distinción:

- La **pseudo-clase :link** se aplica a los vínculos que aún no han sido visitados.
- La **pseudo-clase :visited** se aplica una vez que el vínculo ha sido visitado por el usuario

En HTML y XHTML la pseudo-clase link se aplica a los elementos **A** con el atributo **href**. Por lo tanto, **a:link** y **:link** tienen el mismo alcance.

CÓDIGO CSS

Frente a las siguientes reglas:

```
a:link { color: green; }
a:visited { color: grey; }
```

CÓDIGO XHTML

Antes de ser visitado el siguiente vínculo será mostrado en color verde, y luego de ser visitado se pondrá gris:

```
<p>Un <a href="http://www.dominio.com/">vínculo</a> a otro sitio.</p>
```

🔗 Ver ejemplo de pseudo-clases :link y :visited (CD > ejemplos > css > selectores > pseudo_clases_link_visited.html)

Pseudo-clases dinámicas :hover, :active y :focus

Las aplicaciones de usuario interactivas también pueden cambiar la apariencia de un elemento en respuesta a acciones del usuario, y CSS proporciona tres pseudo-clases dinámicas (es decir, que cambian en respuesta a las acciones del usuario):

- La **pseudo-clase :hover** se aplica cuando el usuario señala un elemento (con algún dispositivo para apuntar) sin activarlo. Por ejemplo, una aplicación de usuario visual cambiar el color de un vínculo al pasar el mouse por encima del mismo. Las aplicaciones de usuario que no ofrecen soporte a los medios interactivos no tienen que apoyar esta pseudo-clase y aquellas con conformidad y que soportan medios interactivos pueden no ser adecuadas para soportar esta pseudo-clase (ej., lápices ópticos).
- La **pseudo-clase :active** se aplica mientras un elemento está siendo activado por el usuario. Por ejemplo, el tiempo en que el usuario mantiene presionado el botón del mouse sobre un vínculo, hasta que lo suelta.
- La **pseudo-clase :focus** se aplica mientras un elemento tiene el foco (acepta eventos del teclado u otras formas de entrada de texto).

Estas pseudo-clases no son mutuamente excluyentes y un elemento puede recibir varias de ellas al mismo tiempo.

Una regla de estilo puede ser descartada por la aplicación de usuario, si la misma establece un tamaño de fuente mayor o cualquier propiedad que requiera redibujar el documento para aceptar las transiciones entre pseudo-clases porque las letras del documento cambian de lugar.

Veamos la sintaxis de estas pseudo-clases:

```
a:link { color: green; }
a:visited { color: gray; }
a:hover { color: blue; }
a:active { color: lime; }
a:focus { background: yellow; }
a:focus:hover { background: gray; }
```

Por las reglas de cascada, para que estas cuatro pseudo-clases funcionen correctamente deben ser especificadas en el orden mostrado en el ejemplo. Observe que `a:hover` debe ir después que las reglas `a:link` y `a:visited`, y `a:active` ubicada después de `a:hover`. Existe una regla muy usada en diseño web que se basa en las palabras love-hate (amor y odio en español) que permiten recordar el orden de las pseudo-clases, ya que se utilizan las letras "L" y "V" de love junto con "H" y "A" de hate, que indican el orden **Link**, **Visited**, **Hover** y **Active**.

El último selector equivale a los elementos a que se encuentran en la pseudo-clase `:focus` y en la pseudo-clase `:hover`.

[🔗 Ver ejemplo de pseudo-clases :hover, :active y :focus \(CD > ejemplos > css > selectores > pseudo_clases_hover_active_focus.html\)](#)

Pseudo-clase de lenguaje :lang

En XHTML es posible especificar el idioma a utilizar en un elemento a través del atributo `lang`, por ejemplo `<p lang="fr">...</p>`. El código para identificar el idioma consta generalmente de dos letras: "es" español, "en" inglés, "de" alemán, "fr" francés, etc. que pueden continuarse con un guión (-) y una subcadena: en-us.

La **pseudo-clase :lang** permite asignar reglas especiales para diferentes lenguajes, de modo de adaptar el texto a las convenciones con respecto al uso de itálicas, sangrías o comillas de cada lenguaje.

El formato es `:lang(C)`, donde el selector tiene equivalencia si se encuentra un elemento con lenguaje "C". Si hay una equivalencia basada en el identificador C sólo o con una subcadena separada por un guión en el valor del elemento, es como si se utilizara el selector de atributos `'lang|=C'`.

La siguiente regla especifica el tipo de comillas que debe usar un elemento `Q` con el atributo `lang=fr` (en

francés):

```
:lang(fr) { quotes: '« ' ' »' }
q:lang(fr) { quotes: '« ' ' »' }
```

La primer regla le aplicará las comillas « » a cualquier elemento en el documento con el lenguaje en francés. Por el contrario, la segunda regla, sólo le aplicará las comillas a los elementos **q** en francés.

🔗 [Ver ejemplo de pseudo-clase :lang \(CD > ejemplos > css > selectores > pseudo_clases_lang.html\)](#)

Pseudo-elementos

Los **pseudo-elementos** permiten aplicar formato a información que está fuera de la estructura del documento. Crean abstracciones acerca de la estructura del documento más allá de las especificadas por el lenguaje del documento, permitiendo hacer referencia a elementos como la primera letra o línea del contenido de un elemento a través de una hoja de estilos. También permiten asignar estilos a un elemento que no existe en el documento fuente (con los pseudo-elementos **:before** y **:after**).

Así como las pseudos-clases, los pseudo-elementos no pueden aparecer en el documento fuente o en la estructura del documento y sólo pueden hacerlo después del sujeto del selector. Los nombres no hacen distinción entre mayúsculas y minúsculas (case-insensitive).

Pseudo-elemento :first-line

El **pseudo-elemento :first-line** aplica un estilo especial a la primera línea de un párrafo. No equivale a ningún elemento HTML real y se corresponde con un pseudo-elemento que las aplicaciones de usuario con conformidad insertarán al comienzo de cada párrafo.

CÓDIGO CSS

```
p:first-line { text-transform: uppercase }
```

Con la regla anterior todas las letras de la primera línea de los párrafos son mostradas en mayúsculas. La medida de esa primera línea está determinada por el tamaño de la fuente, el ancho de la ventana, etc.

CÓDIGO XHTML

De este modo, en un párrafo XHTML como:

```
<p>Duis nonsed tatum er ametuerit, velesenibh euis nit velenim enisci blam ipit
eugait vel eugiam, commodolenim zrillaoreet ing exero exerit ad er am non henit
lortin hent at. Ut lortion equamet nim et non vulput wisi tem aute do corper
sustisisis et dolobore molor sis eugait ipisl diamcon ullam ad magna feumsandit
dolobore ero od molorper ip ex elit amcommolor sent lutatet praessi.</p>
```

RESULTADO

Se tomará la primera línea para mostrarse de la siguiente forma:

```
DUIS NONSED TATUM ER AMETUERIT, VELESENIBH EUIS NIT
velenim enisci blam ipit eugait vel eugiam, commodolenim zrillaoreet
ing exero exerit ad er am non henit lortin hent at. Ut lortion equamet nim
et non vulput wisi tem aute do corper sustisisis et dolobore molor sis
eugait ipisl diamcon ullam ad magna feumsandit dolobore ero od
molorper ip ex elit amcommolor sent lutatet praessi.
```

El código será "re-escrito" por las aplicaciones de usuario para incluir la secuencia ficticia de marcas para **:first-line**, lo que ayuda a mostrar cómo se heredan las propiedades.


```
<p><p:first-line>Duis nonsed tatum er ametuerit, velesenibh euis nit</p:first-line>
velenim enisci blam ipit eugait vel eugiam, commodolenim zrrillaoreet ing exero
exerit ad er am non henit lortin hent at. Ut lortion equamet nim et non vulput
wisi tem aute do corper sustisisis et dolobore molor sis eugait ipisl diamcon
ullam ad magna feumsandit dolobore ero od molorper ip ex elit amcommolor sent
lutatet praessi.</p>
```

🔗 [Ver ejemplo de pseudo-elemento :first-line \(CD > ejemplos > css > selectores > pseudo_elementos_firstline.html\)](#)

El pseudo-elemento `:first-line` es similar a un elemento a nivel de línea, pero con ciertas restricciones y sólo puede ser aplicado en elementos a nivel de bloque, título de tablas (**caption**) o celdas de tablas.

Únicamente las siguientes propiedades se aplican al pseudo-elemento `:first-line`: propiedades de fuentes, propiedades de color, propiedades del fondo, 'espacio entre palabras' (word-spacing), 'espacio entre letras' (letter-spacing), 'decoración del texto' (text-decoration), 'alineación vertical' (vertical-align), 'transformaciones del texto' (text-transform), 'ancho de línea' (line-height).

Pseudo-elemento `:first-letter`

El **pseudo-elemento `:first-letter`** puede ser usado para las "capitulares" y "capitulares caídas", que son efectos tipográficos de uso frecuente. Esta clase de letra inicial es similar a un elemento a nivel de línea cuya propiedad **float** es **none**, de otro modo es similar a un elemento flotante.

Estas son las propiedades que se aplican a los pseudo-elementos `:first-letter`: propiedades de fuentes, 'decoración del texto' (text-decoration), 'transformaciones del texto' (text-transform), 'espacio entre letras' (letter-spacing), 'espacio entre palabras' (word-spacing) (cuando sea apropiado), 'ancho de línea' (line-height), 'flotante' (float), 'alineación vertical' (vertical-align) (solo si 'float' es 'none'), propiedades de los márgenes, propiedades de relleno, propiedades de los bordes, propiedades de color, propiedades del fondo.

Los pseudo-elementos `:first-letter` se aplican a bloques, listas de objetos (list-item), celdas de tablas (table-cell), títulos de tablas (table-caption) y en elementos de bloques en línea (inlineblock).

Si un elemento es una lista de elementos ('display: list-item'), `:first-letter` se aplica a la primera letra en la caja principal después del marcador.

Si un elemento tiene el contenido '`:before`' o '`:after`', `:first-letter` se aplica a la primera letra del elemento incluido en ese contenido. Ej., después de la regla `p:before {content: "Note: "}`, el selector `p:first-letter` se aplica a la "N" de "Note".

Ejemplos

Aplicación 1

CÓDIGO CSS

Veamos la creación de una capitular:

```
p { line-height: 1.1 }
p:first-letter { font-size: 3em; font-weight: bold; color: gray; }
```

CÓDIGO XHTML

```
<p>Duis nonsed tatum er ametuerit, velesenibh euis nit velenim enisci blam ipit
eugait vel eugiam, commodolenim zrrillaoreet ing exero exerit ad er am non henit
lortin hent at. Ut lortion equamet nim et non vulput wisi tem aute do corper
sustisisis et dolobore molor sis eugait ipisl diamcon ullam ad magna feumsandit
dolobore ero od molorper ip ex elit amcommolor sent lutatet praessi.</p>
```

RESULTADO

Esto daría como resultado:

Duis nonsed tatum er ametuerit, velesenibh euis nit velenim enisci blam ipit eugait vel eugiam, commodolenim zrillaoreet ing exero exerit ad er am non henit lortin hent at. Ut lortion equamet nim et non vulput wisi tem aute do corper sustisisis et dolobore molor sis eugait ipisl diamcon ullam ad magna feumsandit dolobore ero od molorper ip ex elit amcommolor sent lutatet praessi.

🔗 [Ver ejemplo resultado de aplicación del pseudo-elemento :first-letter \(CD > ejemplos > css > selectores > pseudo_elementos_firstletter.html\)](#)

Aplicación 2

CÓDIGO CSS

El siguiente código creará una capitular caída que abarca dos líneas:

```
p { line-height: 12px; }
p:first-letter {font-size: 260%; font-weight: bold; float: left; }
span { text-transform: uppercase; }
```

CÓDIGO XHTML

```
<p><span>Duis nonsed tatum er ametuerit</span>, velesenibh euis nit velenim enisci
blam ipit eugait vel eugiam, commodolenim zrillaoreet ing exero exerit ad er
am non henit lortin hent at. Ut lortion equamet nim et non vulput wisi tem aute
do corper sustisisis et dolobore molor sis eugait ipisl diamcon ullam ad magna
feumsandit dolobore ero od molorper ip ex elit amcommolor sent lutatet praessi.</
p>
```

RESULTADO

Lo que daría como resultado:

DUIS NONSED TATUM ER AMETUERIT, velesenibh euis nit velenim enisci blam ipit eugait vel eugiam, commodolenim zrillaoreet ing exero exerit ad er am non henit lortin hent at. Ut lortion equamet nim et non vulput wisi tem aute do corper sustisisis et dolobore molor sis eugait ipisl diamcon ullam ad magna feumsandit dolobore ero od molorper ip ex elit amcommolor sent lutatet praessi.

🔗 [Ver ejemplo resultado de aplicación del pseudo-elemento :first-letter \(CD > ejemplos > css > selectores > pseudo_elementos_firstletter1.html\)](#)

Aplicación 3

Con el propósito de conseguir el formato de las capitulares caídas tradicionales, las aplicaciones de usuario pueden aproximar los tamaños de la fuente para emparejar las líneas base.

La puntuación que precede a la primera letra debe ser incluida, incluso si la primera letra es un número. Ej. "5 millones de pesos repartidos por Lotería Nacional el pasado miércoles".

"5 0 millones de pesos repartidos por Lotería Nacional el pasado miércoles".

🔗 [Ver ejemplo resultado de aplicación del pseudo-elemento :first-letter \(CD > ejemplos > css > selectores >](#)

pseudo_elementos_firstletter2.html

Pseudo-elementos :before y :after

Con los **pseudo-elementos :before (antes) y :after (después)** se puede insertar un contenido antes o después de un elemento determinado y definir el estilo del contenido insertado. La propiedad '**content**', junto con estos pseudo-elementos, especifican lo que se inserta.

Podemos insertar antes de cada elemento **H1** el texto "CAPÍTULO" en mayúsculas y verde y antes de cada **H2** el texto "Tema" sin necesidad de escribirlo en cada encabezado. También podemos hacer que cada capítulo termine con un icono especial.

```
h1:before {content: "Capítulo: "; text-transform: uppercase; color: green; }  
h2:before {content: "Tema: "; color: blue; }  
body:after {content: url("icono.gif")}
```

[🔗 Ver ejemplo y resultado de aplicación de los pseudo-elementos :before y :after \(CD > ejemplos > css > selectores > pseudo_elementos_before_after.html\)](#)

4. Asignación de valores, cascada, y herencia



VALORES ESPECIFICADOS, COMPUTADOS, USADOS Y REALES

Una aplicación de usuario debe construir la estructura del documento, y luego asignar a cada elemento de la estructura un valor por cada propiedad que le es asignada.

El valor final de una propiedad es el resultado de un cálculo en 4 pasos: el valor se determina por medio de la especificación ("**valor especificado**"), luego resuelto a un valor que se utilice para la herencia ("**valor computado**"), después se convierte en un valor absoluto en caso de necesidad ("**valor usado**"), y finalmente se transforma según las limitaciones del entorno local ("**valor real**").

Valores especificados

Todas las propiedades tienen siempre un valor asignado. El **valor especificado** puede ser:

- el especificado en la CSS o determinado por la cascada,
- el valor heredado (si el elemento no es el raíz -html- es el computado del elemento padre),
- o el valor inicial de la propiedad (predeterminado para cada propiedad).

Valores computados

Los valores especificados se resuelven a los **valores computados** durante la cascada. Así las URIs relativas válidas se hacen absolutas y las unidades relativas ('em', 'ex' y porcentajes) se computan a píxel o a medidas absolutas.

El valor computado de URIs no válidas y absolutas es el valor especificado.

Cuando el valor especificado no es 'heredado', el valor computado de una propiedad se determina según lo especificado

en la definición de la propiedad como Valor Computado. (Ver "[Herencia](#)" para la definición de valores computados cuando el valor especificado es 'heredado')

El valor computado existe incluso cuando la propiedad no se aplica. Sin embargo, algunas propiedades pueden definir el valor computado de una propiedad para un elemento que dependa o no de la propiedad aplicada a ese elemento.

CÓDIGO CSS

Cuando se especifica un valor relativo es necesario hacer un cálculo para obtener su valor absoluto. Teniendo el siguiente código:

```
p { font-size: 10px; }
strong { font-size: 115%; }
```

CÓDIGO XHTML

```
<p>Lore mincing euguer <strong>aliquat ilit illaor alisl <em>delenibh</em> eu  
feuis dit wisissenis nullutat</strong>. Duip erat dip ea core feuisi. </p>
```

Para resolver el 115% de **STRONG**, hay que tomar el valor del elemento **P** que es 11.5px, lo que daría un valor computado del **STRONG** de 11.5px (10x115%=11.5px). Los hijos de **STRONG**, en este caso el **EM**, no heredan el valor especificado (115%) sino el valor computado (11.5px).

Valores usados

Los valores computados son procesados generalmente sin dar formato al documento, pero en ciertas ocasiones sólo pueden determinarse una vez que el documento está siendo presentado. Por ejemplo, el ancho de un elemento puede establecerse para ser un 50% de su bloque contenedor, en cuyo caso el ancho no puede determinarse hasta que no se haya determinado el ancho de su contenedor.

Entonces, el **valor usado** es el resultado de tomar el valor computado y resolver esas dependencias en un valor absoluto.

Valores reales

Un valor usado es en principio el valor usado para procesar el documento, pero puede suceder que el valor resultante no pueda ser utilizado por una aplicación de usuario debido a limitaciones técnicas, y el mismo deba ser aproximado a otro valor. El **valor real** es el valor usado después de que se han aplicados las aproximaciones.

En el ejemplo anterior, no sería posible procesar una fuente con un tamaño de 11.5px, por lo tanto el navegador deberá ajustarlo a 11px o 12px. Y como esto, una aplicación de usuario no podría procesar bordes con un ancho que no esté expresado en píxeles enteros.

HERENCIA

Algunos valores son heredados por los hijos de un elemento en la estructura del documento y cada propiedad define específicamente si es heredada o no.

Veamos un elemento **H1** con un texto enfatizado con **EM** en su interior:

```
<h1>El título <em>que tienen una itálica</em> es importante!</h1 >
```

Si no se ha asignado ningún color al elemento **EM**, el mismo heredará el color del elemento padre (si el **H1** es rojo, entonces el **EM** también lo será).

Cuando ocurre la herencia, los elementos heredan valores computados y el valor computado del elemento padre se convierte, a la vez, en el valor especificado y computado en el hijo.

CÓDIGO CSS

Volviendo a un ejemplo anterior:

```
p { font-size: 10px; }
strong { font-size: 115%; }
```

CÓDIGO XHTML

```
<p>Lore mincing euguer <strong>aliquat ilit illaor alisl <em>delenibh</em> eu
feuis dit wisissenis nullutat</strong>. Duip erat dip ea core feuisi. </p>
```

La propiedad `'font-size'` del elemento **STRONG** será de 11.5px (10px x 115%= 11.5px). Como el valor es heredado, el elemento **EM** también tendrá 11.5px. Si la aplicación de usuario no puede procesar la fuente con decimales, entonces el valor real para **STRONG** será de 11px (o 12px) y por lo tanto también lo será para el **EM**.

El valor 'inherit' (heredado)

Cada propiedad puede tener también un valor especificado como `'inherit'` (heredado), con lo cuál la propiedad de un elemento toma el mismo valor que el elemento padre. El valor `'inherit'` puede ser utilizado para reforzar los valores heredados, y sobre propiedades que normalmente no se heredan.

Por ejemplo, un usuario puede declarar en su hoja de estilo que todos los elementos **P** hereden el color negro de la tipografía del cuerpo, de modo de sobrescribir cualquier cambio de color establecido por el autor. El uso de `!important` obliga a que se apliquen las reglas del usuario.

```
body {
  color: black !important;
  background: white !important;
}
p {
  color: inherit !important;
  background: inherit !important;
}
```

LA REGLA @IMPORT

La regla `'@import'` permite importar hojas de estilo desde otras hojas de estilo y debe preceder a todas las reglas especificadas en una hoja de estilo. La palabra clave `'@import'` puede ir seguida por una cadena con el URI de la hoja de estilo a incluir o por la construcción `url(...)` dentro de cuyos paréntesis se especifica la URL.

Las líneas siguientes son equivalentes:

```
@import "estilo.css";
@import url("estilo.css");
```

A cada regla `@import` se le puede especificar el medio al cuál están destinadas, para que las aplicaciones de usuario no levanten hojas de estilos que no soportan. Los medios deben ser especificados luego del URI y separados por comas:

```
@import url("impresion.css") print;
@import url("proyeccion.css") projection, tv;
```

La ausencia de medios o la colocación del medio `'all'` indican que la importación es incondicional.

CASCADA

Las hojas de estilo pueden tener tres orígenes diferentes: el autor, el usuario y la aplicación de usuario.

- **Autor:** El autor especifica las hojas de estilo para un documento según las convenciones del lenguaje del documento, para dar a dicho documento un estilo específico.
- **Usuario:** El usuario puede especificar una hoja de estilo con información de estilo para un documento particular. Por ejemplo, para mostrar las tipografías de un mayor tamaño o en otro color cuando el especificado por el autor no es legible para el usuario.
- **Aplicaciones de usuario:** Las aplicaciones de usuario con conformidad deben aplicar una hoja de estilo predeterminada previa a todas las hojas de estilo (de autores y usuarios) para un documento. La hoja de estilo debe presentar los elementos del lenguaje del documento de modo que sean coherente con lo esperado en cuanto a la presentación del lenguaje del documento (Ej. el elemento EM se presenta usando una fuente itálica). (Ver [Anexo > Hoja de estilo por defecto para HTML 4.0](#))

Estas hojas de estilo con tres orígenes se superponen en su acción e interactúan de acuerdo con la cascada, que asigna un peso o importancia a cada regla de estilo, siendo la de mayor peso la que tiene preponderancia.

Por defecto, las reglas en las hojas de estilo del autor tienen más peso que las reglas en las hojas de estilo del usuario. La preponderancia (precedencia) se invierte, sin embargo, con la regla `!important`. Todas las reglas del usuario y del autor tienen más peso que las reglas de la hoja de estilo predeterminada en la AU (Aplicación de usuario).

Orden de cascada

Para encontrar el valor para una combinación de elemento/propiedad, las aplicaciones de usuario deben aplicar el siguiente orden de disposición:

1. Encontrar todas las declaraciones que se aplican a un elemento y a la propiedad en cuestión, para el tipo de medio al que está dirigido. Las declaraciones se aplican si el selector asociado coincide con el elemento en cuestión.
2. Clasificar por la importancia (normal o importante) y por el origen (autor, usuario, o aplicación de usuario). En orden ascendente:
 - hojas de estilo de la aplicación de usuario
 - hojas de estilo normales del usuario
 - hojas de estilo normales del autor
 - hojas de estilo importantes del autor
 - hojas de estilo importantes del usuario
3. Clasificar por la especificidad del selector: Los selectores más específicos sustituirán a los más generales. Los Pseudo-elementos y las pseudo-clases se cuentan como elementos y clases normales, respectivamente.
4. Finalmente, clasificar por el orden especificado: si dos reglas tienen el mismo peso, origen y especificidad, la última en ser especificada se aplica. Las reglas en hojas de estilo importadas se consideran que están antes de cualquier regla en la propia hoja de estilo.

La definición de `!important` en declaraciones individuales en la hoja de estilo del autor, da a las mismas un peso más relevante que las del lector. Es por lo tanto importante que la aplicación de usuario le brinde al usuario la posibilidad de neutralizar la influencia de determinada hoja de estilo, Ej. a través de un menú desplegable.

Las reglas `!important`

Por defecto, las reglas de la hoja de estilo del autor sustituyen las de la hoja de estilo del usuario. Sin embargo, una

declaración `!important` a continuación de la declaración hace que esta tome precedencia sobre una declaración normal. Las hojas de estilo del autor y del usuario pueden contener declaraciones `!important`, y en ese caso las reglas `!important` del usuario sustituyen a las reglas `!important` del autor.

Esta propiedad del CSS brinda a los usuarios con requisitos especiales (fuentes grandes, combinaciones de color, etc.) el control sobre la presentación.

Hay que tener en cuenta que al declarar una propiedad resumida (Ej. `'background'`) como `!important` se declaran todas sus sub-propiedades como `!important`.

CSS DEL USUARIO

Veamos los siguientes ejemplos de aplicación:

```
/* Hoja de estilo del usuario */
P { text-indent: 1em ! important }
P { font-style: italic ! important }
P { font-size: 18pt }
```

CSS DEL AUTOR

```
/* Hoja de estilo del autor */
P { text-indent: 1.5em !important }
P { font: 12pt sans-serif !important }
P { font-size: 24pt }
```

La primera regla en la hoja de estilo del usuario contiene una declaración `!important`, que sustituye a la correspondiente declaración en la hoja de estilo del autor.

La segunda declaración también sustituye a la del autor debido a que ha sido marcada como `!important`.

Sin embargo, la tercera regla en la hoja de estilo del usuario no es `!important` y entonces no tendrá influencia frente a la segunda regla en la hoja de estilo del autor (que definió el estilo en una propiedad resumida).

Asimismo, la tercera regla del autor perderá con la segunda regla del autor debido a que la segunda regla es `!important`. Esto muestra que las declaraciones `!important` tienen la misma función también dentro de las hojas de estilo del autor.

Cálculo de la especificidad de un selector

Si bien existen métodos de cálculo de la especificidad de un selector, en términos generales y de mayor a menor especificidad tenemos los siguientes selectores:

- el atributo `style=""` colocado en línea dentro de un elemento XHTML,
- un selector de id (Ej. `#miid`),
- un selector de clase (Ej. `.miclase`),
- selectores de atributos (Ej. `h1[title]`),
- selectores de hijos, descendientes, hermanos adyacentes. (Ej. `ul li ol`),
- selectores de tipo (Ej. `li`) y
- selector universal (`*`).

En el siguiente ejemplo, el color del párrafo será verde, porque el elemento atributo `style` en línea tiene mayor preponderancia que el `id` declarado en el elemento `STYLE`.


```
<head>
  <style type="text/css">
    #miid { color: red }
  </style>
</head>
<body>
  <p id="miid" style="color: green">texto</p>
</body>
```

5. Tipos de medios



INTRODUCCIÓN A LOS TIPOS DE MEDIOS

Las hojas de estilo especifican cómo debe ser presentado un documento en diferentes medios: en la pantalla, en el papel, con un sintetizador de voz, con un dispositivo braille, etc.

Algunas propiedades CSS se diseñan solamente para ciertos medios (Ej. como `'page-break-before'` para medios paginados) y otras pueden compartirse en hojas de estilo para diferentes medios, pero requieren de distintos valores según el medio. Por ejemplo, `'font-size'` es útil para pantalla e impresión, pero esta propiedad deberá tener un valor mayor para la pantalla.

Frente a esto, en ciertas ocasiones es necesario expresar que una hoja de estilo o una sección de la misma se aplican a un tipo de medio específico.

ESPECIFICACIÓN DE HOJAS DE ESTILO DEPENDIENTES DE LOS MEDIOS

Hay dos maneras de especificar las dependencias de los medios para las hojas de estilo:

- Especificar el medio de destino dentro de una hoja de estilo con las reglas arroba: `@media` o `@import`.

```
@import url("principal.css") screen;
@media print {
  /* las reglas para impresión van aquí. */
}
```

- Especificar el medio de destino dentro del lenguaje del documento con el atributo `"media"`, al vincular una hoja de estilo externa con el elemento `LINK`.

```
<head>
<title>Vínculo a un medio</title>
<link rel="stylesheet" type="text/css" media="print, handheld"
href="principal.css">
</head>
```

La regla @media

Una regla **@media** especifica los tipos de medios de destino (separados por comas) de un sistema de reglas (delimitadas por llaves). Esta regla permite colocar en una misma hoja de estilo reglas para diferentes medios:

```
@media print {
body { font-size: 10pt }
}
@media screen {
body { font-size: 13px }
}
@media screen, print {
body { line-height: 1.2 }
}
```

TIPOS DE MEDIOS RECONOCIDOS

Los nombres de los tipos de medios de CSS reflejan los dispositivos de destino para los cuales las propiedades serán aplicadas.

Nota: las descripciones entre paréntesis sólo brindan una aproximación a la clase de dispositivo al cuál el tipo de medio hace referencia.

Tipo	Explicación
screen	Para pantallas no paginadas de computadora.
tty	Para medios que utilicen una cuadrícula de caracteres de ancho fijo, como teletipos, terminales y dispositivos portátiles con posibilidades limitadas de representación.
tv	Para dispositivos tipo televisión (baja resolución, en color, desplazamiento limitado).
projection	Para proyectores.
handheld	Para dispositivos de mano (pantalla pequeña, monocromos, gráficos por mapas de bits, ancho de banda limitado).
print	Para material paginado, opaco, y para documentos que se ven en una pantalla en modo de presentación preliminar a la impresión.
braille	Para dispositivos táctiles braille.
aural	Para sintetizadores de voz.
all	Apropiado para todos los dispositivos.
embossed	Previsto para las impresoras de páginas braille.
speech	Previsto para los sintetizadores de voz

Los nombres de los tipos de medios no diferencian entre mayúsculas y minúsculas (case-insensitive).

Los tipos de medios son mutuamente excluyentes, es decir que una aplicación del usuario puede soportar solamente un tipo de medio al interpretar un documento. Sin embargo, las aplicaciones de usuario pueden tener diversos modos que soporten diferentes tipos de medios (al previsualizar impresión, las UA hacen uso de las hojas de estilo para impresión).

Cuando hay un tipo de medio desconocido, la regla **@media** es ignorada.

Grupos de medios

Debido a que las propiedades se aplican generalmente a varios tipos de medios, al definir cada propiedad, en la columna “**Medio**” se listan los grupos de medios en vez de los tipos de medio individuales.

Se definen los siguientes grupos de medios:

- Continuos (continuous) o **paginados (paged)** .
- **Visual (visual), audio (audio), voz (speech), o táctil (tactile).**
- **Rejilla (gris)** (para dispositivos de rejilla de caracteres), o **mapa de bit (bitmap).**
- **Interactivos (interactive)** (para los dispositivos que permiten la interacción con el usuario), o **estáticos (static)** (para los que no lo hacen).
- **Todos (all)** (incluye todos los tipos de medios)

La tabla siguiente muestra las relaciones entre los grupos de medios y los tipos de medios:

Relación entre los grupos de medios y los tipos de medios				
Tipos de medios	Grupos de medios			
	continuo/paginado	visual/audio/voz/táctil	grilla/bitmap	interactivo/estático
braille	continuo	táctil	grilla	ambos
embossed	paginado	táctil	grilla	estático
handheld	ambos	visual, audio, voz	ambos	ambos
print	paginado	visual	bitmap	estático
projection	paginado	visual	bitmap	interactivo
screen	continuo	visual, audio	bitmap	ambos
speech	continuo	voz	N/A	ambos
tty	continuo	visual	grilla	ambos
tv	ambos	visual, audio	bitmap	ambos

6. Modelo de cajas

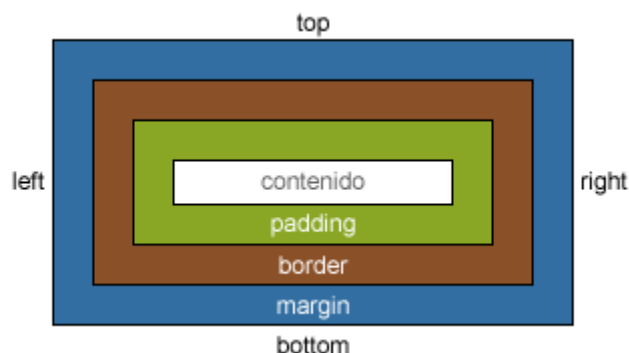


El modelo de caja de CSS describe las cajas rectangulares que se generan para todos los elementos en la estructura del documento y se presentan según el [modelo de formato visual](#).

DIMENSIONES DE LA CAJA

Cada caja tiene un área de **contenido** (texto, imagen, etc.) y las áreas circundantes opcionales de **padding** (relleno), **border** (borde) y **margin** (margen).

El diagrama siguiente muestra cómo se relacionan:



El margen (margin), el borde (border), y el relleno (padding) pueden ser divididos en los segmentos superior (top), derecho (right), inferior (bottom) e izquierdo (left).

Las dimensiones del área de contenido de una caja (el ancho y alto del contenido) depende de varios factores: si el elemento tiene asignada las propiedades `'width'` (ancho) o `'height'` (alto), si la caja contiene texto u otras cajas, si la caja es una tabla, etc.

El estilo del fondo del contenido y el relleno (padding) está determinado por la propiedad `'background'` del elemento. El fondo del borde está determinado por las propiedades del border y el fondo del margen es siempre transparente.

EJEMPLO DE MÁRGENES, RELLENOS Y BORDES

Este ejemplo ilustra cómo interactúan los márgenes, los rellenos y los bordes en un documento con un elemento **UL** que contiene a dos **LI**:

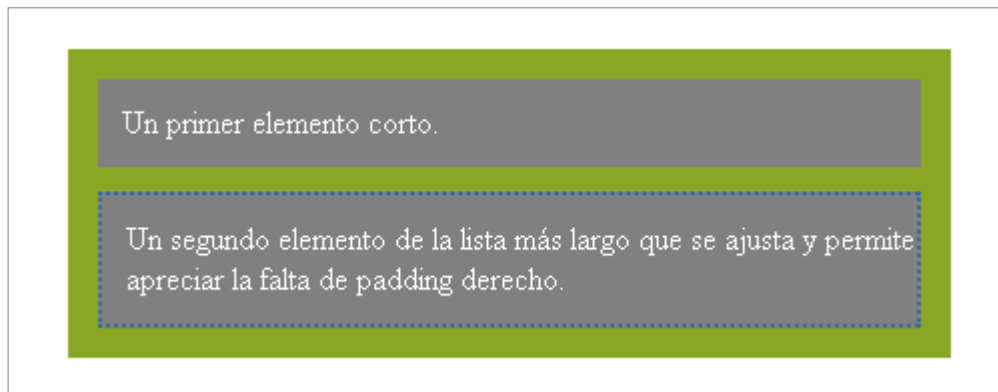
CÓDIGO

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba con margin, padding y border</title>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
  <meta name="author" content="Maira Vykus"/>
  <style type="text/css">
    ul {
      width: 430px;
      background: #89A725;
      margin: 12px;
      padding: 3px;
    }
    li {
      color: white;
      background: gray;
      margin: 12px;
      padding: 12px 0px 12px 12px;
      list-style: none;
    }
    li.border {
      border: 2px dotted #326EA1;
    }
  </style>
</head>

<body>
  <ul>
    <li>Un primer elemento corto.</li>
    <li class="border">Un segundo elemento de la lista más largo que se ajusta y
    permite apreciar la falta de padding derecho.</li>
  </ul>
</body>
</html>
```

RESULTADO

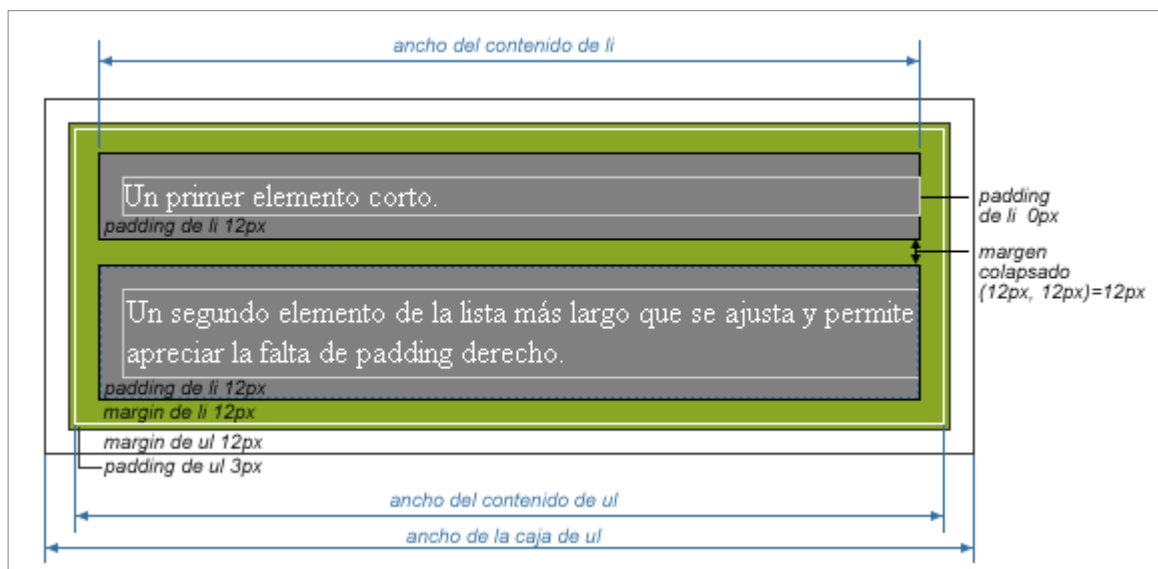
Esta imagen muestra el resultado que produciría el código anterior:



🔗 [Ver página con la aplicación de márgenes, rellenos y bordes \(CD > ejemplos > css > caja > todos.html\)](#)

EXPLICACIÓN

La siguiente ilustra la relación entre los márgenes, el relleno y los bordes del elemento **UL** y de sus elementos hijos **LI**.



En base a esta imagen podemos observar que:

- El ancho del contenido para cada caja de **LI** es calculado de arriba hacia abajo; el bloque de contención de cada caja **LI** es establecido por el elemento **UL**.
- El alto de cada **LI** depende del alto del contenido, más el relleno, bordes y márgenes superiores e inferiores. Los márgenes verticales entre las cajas **LI** se colapsan o cierran.
- El relleno (padding) de los **LI** se ha fijado en 0, lo que se hace evidente en el segundo **LI**.
- Los márgenes de las cajas **LI** son transparentes y por ello se observa a través de ellos el color verde del fondo del **UL**.
- El segundo elemento **LI** especifica un borde punteado (la propiedad '`border-style`' con valor '`dotted`').

PROPIEDADES DEL MARGEN (MARGIN)

Las propiedades del margen especifican el ancho del área del margen de una caja, es decir, el espacio alrededor del elemento. Los márgenes superior, derecho, inferior e izquierdo pueden ser cambiados independientemente usando propiedades separadas. La propiedad resumida '`margin`' establece el margen para los cuatro lados al mismo tiempo. Estas propiedades se aplican a todos los elementos, pero los márgenes verticales no tienen ningún efecto en los elementos en línea (inline) no reemplazados. Se pueden usar valores negativos (por ejemplo, `margin-top: -8px`), pero puede haber límites específicos de la implementación.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'margin-right' 'margin-left'	<ancho-del-margen> inherit	0	Todos los elementos, excepto aquellos elementos de tabla con display diferente de table-caption, table e inline-table	No	Se refiere al ancho del bloque de contención	Visual	1
'margin-top' 'margin-bottom'	<ancho-del-margen> inherit						
'margin'	<ancho-del-margen> {1,4} inherit						
Valor computado:	En las cinco propiedades es el porcentaje especificado o la medida absoluta.						

Valores posibles para `<ancho-del-margen>` (Más información en: "[Valores](#)"):

Nombre	Explicación
<code><medida></code>	Especifica un ancho fijo. El valor por defecto es 0.
<code><porcentaje></code>	El porcentaje se calcula con respecto al ancho del <u>bloque de contención</u> de la caja generada.
auto	El valor es determinado por el navegador.

Como ya expresamos, la propiedad '`margin`' es una propiedad resumida para fijar '`margin-top`', '`margin-right`', '`margin-bottom`', y '`margin-left`' en un sólo paso. Los signos {1,4} significan que pueden especificarse de 1 a 4 valores para `<ancho-del-margen>`. Cuando hay:

- Un sólo valor: el margen se aplica con ese valor a todos los lados (Ej. `margin: 2em;`)
- Dos valores: los márgenes superior e inferior son determinados por el primer valor y los márgenes derecho e izquierdo por el segundo (Ej. `margin: 2em 4em;`)
- Tres valores: el margen superior es definido por el primer valor, el izquierdo y el derecho por el segundo, y el inferior por el tercero (Ej. `margin: 2em 4em 5em;`)
- Cuatro valores: se aplican al margen superior, derecho, inferior e izquierdo, respectivamente (Ej. `margin: 2em 4em 5em 1em;`)

Ejemplos

A continuación podemos ver las cuatro formas posibles de conglomerar en la propiedad '`margin`' los cuatro lados:

```
h1 { margin: 2em } /* todos los márgenes en 2em */
h1 { margin: 1em 2em } /* top y bottom = 1em, right y left = 2em */
h1 { margin: 1em 2em 3em } /* top = 1em, right y left = 2em, bottom = 3em */
h1 { margin: 1em 2em 3em 5em } /* top=1em, right=2em, bottom=3em, left=5em */
```

La última regla anterior, podría haberse escrito de la siguiente forma:


```
h1 {
margin-top: 1em;
margin-right: 2em;
margin-bottom: 3em;
margin-left: 5em;
}
```

- 🔗 Todas las propiedades de margin en una sola declaración (CD > ejemplos > css > caja > margin_01.html)
- 🔗 Asignación del margen superior en píxeles (CD > ejemplos > css > caja > margin_02.html)
- 🔗 Asignación del margen inferior en porcentajes (CD > ejemplos > css > caja > margin_03.html)

Márgenes cerrados

Márgenes cerrados significa que los márgenes de dos o más cajas (que pueden estar una al lado de la otra o anidadas) se combinan para formar un solo margen. Los márgenes horizontales nunca se cierran, pero los márgenes verticales pueden cerrarse o combinarse entre ciertas cajas:

- Dos o más márgenes verticales adyacentes de cajas de bloques en el flujo normal se cierran. El ancho del margen resultante es el máximo de los anchos de los márgenes adyacentes. En el caso de márgenes negativos, el máximo absoluto de los márgenes adyacentes negativos es restado del máximo de los márgenes adyacentes positivos. Si no hay ningún margen positivo, el máximo absoluto de los márgenes adyacentes negativos es cero.
- Si los márgenes superiores (top) e inferiores (bottom) de una caja son adyacentes, entonces es posible para los márgenes cerrarse con ella. En este caso, la posición del elemento depende de su relación con los otros elementos cuyos márgenes están cerrados.
- Si los márgenes del elemento están cerrados con el margen superior del padre, el límite del borde superior de la caja es definido para ser el mismo que el del padre.
- Si no, o el padre del elemento no participa en el cierre del margen, o solamente el margen inferior (bottom) del padre está implicado. La posición del límite del borde superior del elemento es la misma que tendría si el elemento tuviera un borde superior no nulo (distinto de cero).

PROPIEDADES DEL RELLENO (PADDING)

Las propiedades del relleno especifican el ancho del área de relleno de una caja, es decir, el espacio entre el borde del elemento y su contenido. La propiedad resumida `'padding'` define el relleno para los cuatro lados mientras que las otras propiedades de relleno sólo definen sus respectivos lados.

A diferencia de las propiedades del margen, los valores del relleno no pueden ser negativos.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'padding-top' 'padding-right' 'padding-bottom' 'padding-left'	<code><padding-width></code> inherit	0	Todos los elementos, excepto table-row-group, table-header-group, table-footer-group, table-row, table-column-group y table-column	No	Se refiere al ancho del bloque de contención	Visual	1
'padding'	<code><padding-width></code> {1,4} inherit						
Valor computado:	Porcentaje especificado o la medida (longitud) absoluta						

Valores posibles para `<padding-width>` (Más información en: "[Valores](#)"):

Nombre	Explicación
<medida>	Especifica un ancho fijo. El valor por defecto es 0.
<porcentaje>	El porcentaje se calcula con respecto al ancho del <u>bloque de contención</u> de la caja generada, aún para 'padding-top' y 'padding-bottom'

La propiedad `'padding'` es una propiedad resumida para definir las propiedades `'padding-top'`, `'padding-right'`, `'padding-bottom'`, y `'padding-left'` en un mismo lugar en la hoja de estilo. Los signos `{1, 4}` significan que pueden especificarse de 1 a 4 valores para `<padding-width>`. Cuando hay:

- Un sólo valor: el padding se aplica con ese valor a todos los lados (Ej. `padding: 2em;`)
- Dos valores: los rellenos superior e inferior son determinados por el primer valor y los rellenos derecho e izquierdo por el segundo (Ej. `padding: 2em 4em;`)
- Tres valores: el relleno superior es definido por el primer valor, el izquierdo y el derecho por el segundo, y el inferior por el tercero (Ej. `padding: 2em 4em 5em;`)
- Cuatro valores: se aplican al relleno superior, derecho, inferior e izquierdo, respectivamente (Ej. `padding: 2em 4em 5em 1em;`)

Ejemplos

El fondo del relleno será igual que el fondo del contenido y está establecido por la propiedad `'background'`:

```
h1 {  
  background: white;  
  padding: 5em;  
}
```

El código de arriba especifica 5em para los rellenos superior, derecho, inferior e izquierdo. La unidad `'em'` es relativa al tamaño de la fuente del elemento y `'1em'` es igual al tamaño de la fuente en uso.

- 🔗 [Todas las propiedades de padding en una declaración \(CD > ejemplos > css > caja > padding_01.html\)](#)
- 🔗 [Asignación del padding izquierdo en píxeles \(CD > ejemplos > css > caja > padding_02.html\)](#)
- 🔗 [Asignación del padding derecho en porcentajes \(CD > ejemplos > css > caja > padding_03.html\)](#)

PROPIEDADES DEL BORDE (BORDER)

Las propiedades del borde aplicables a todos los elementos, especifican el ancho, color y estilo del área del borde de una caja.

Ancho del borde: `'border-top-width'`, `'border-right-width'`, `'borderbottom-width'`, `'border-left-width'` y `'border-width'`

Las propiedades del ancho del borde especifican el ancho del área de los bordes superior (top), derecho (right), inferior (bottom) e izquierdo (left) de una caja.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'border-top-width' 'border-right-width' 'border-bottom-width' 'border-left-width'	<border-width> inherit	medium	Todos los elementos.	No	N/A	Visual	1
'border-width'	<border-width>{1,4} inherit						
Valor computado:	El valor computado es la medida absoluta; '0' si el estilo del borde es 'none' o 'hidden'						

Valores posibles para **<border-width>** (Más información en: "[Valores](#)"):

Nombre	Explicación	
<medida>	El grosor del borde tiene un valor específico. Las dimensiones del borde explícitas no pueden ser negativas.	
thin	Un borde fino.	Su interpretación, que debe mantenerse constante en todo el documento, depende de la aplicación de usuario, pero siempre se debe tener en cuenta la siguiente relación: 'thin' <= 'medium' <= 'thick'
medium	Un borde mediano.	
thick	Un borde grueso.	

La propiedad 'border-width' resume 'border-top-width', 'border-right-width', 'border-bottom-width' y 'border-left-width' en un mismo lugar. Los signos {1,4} significan que pueden especificarse de 1 a 4 valores para <border-width>. Cuando hay:

- Un sólo valor: se aplica con ese valor a todos los lados (Ej. `border-width: 2em;`)
- Dos valores: los bordes superior e inferior son determinados por el primer valor y los bordes derecho e izquierdo por el segundo (Ej. `border-width: 2em 4em;`)
- Tres valores: el borde superior es definido por el primer valor, el izquierdo y el derecho por el segundo, y el inferior por el tercero (Ej. `border-width: 2em 4em 5em;`)
- Cuatro valores: se aplican al borde superior, derecho, inferior e izquierdo, respectivamente (Ej. `border-width: 2em 4em 5em 1em;`)

En los ejemplos siguientes se puede observar la aplicación de diferentes bordes a un elemento H1.

```
h1 { border-width: thin } /* thin thin thin thin */
h1 { border-width: thin thick } /* thin thick thin thick */
h1 { border-width: thin thick medium } /* thin thick medium thick */
```

Color del Borde: 'border-top-color', 'border-right-color', 'border-bottom-color', 'border-left-color' y 'border-color'

Las propiedades del color del borde especifican el color del borde de la caja.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'border-top-color' 'border-right-color' 'border-bottom-color' 'border-left-color'	<color> transparent inherit	el valor de la propiedad 'color'	Todos los elementos.	No	N/A	Visual	2
'border-color'	[<color> transparent]{1,4} inherit						1
Valor computado:	Cuando lo toma de la propiedad 'color', el valor computado es 'color'; sino, el especificado						

Valores posibles para **<color>** (Más información en: “Valores”):

Nombre	Explicación
<color>	Especifica un valor de color.
transparent	El borde es transparente (no obstante puede tener grosor).

La propiedad '**border-color**' puede tener de uno a cuatro valores, y los valores son aplicados a los distintos lados como en '**border-width**'.

Si el color del borde de un elemento no es especificado con una propiedad de borde, las aplicaciones de usuario deben tomar el valor de la propiedad '**color**' del elemento como el valor computado para el color del borde.

En cualquiera de los dos ejemplos el resultado será una línea roja:









```
p {
  color: red;
  background: white;
  border: solid;
}
p.prueba {
  border: 1px solid red;
}
```

Estilo del borde: 'border-top-style', 'border-right-style', 'border-bottom-style', 'border-left-style' y 'border-style'

Las propiedades del estilo del borde especifican el estilo de la línea del borde de una caja (sólida, doble, punteada, etc.).

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'border-top-style' 'border-right-style' 'border-bottom-style' 'border-left-style'	<border-style> inherit	none	Todos los elementos.	No	N/A	Visual	2
'border-style'	<border-style>{1,4} inherit						1
Valor computado:	Como el especificado						

Valores posibles para **<border_style>** (Más información en: “Valores”):

Nombre	Explicación	
none	ningún borde; el ancho del borde es cero	
hidden	Igual que 'none' , excepto en términos de resolución de conflictos de bordes para los elementos de tablas.	
dotted	El borde es una serie de puntos.	
dashed	El borde es una serie de pequeños segmentos de líneas.	
solid	El borde es un único segmento de línea.	
double	El borde son dos líneas sólidas. La suma de las dos líneas y el espacio entre ellas es igual al valor de 'border-width' .	
groove	El borde luce como si estuviera tallado en el lienzo.	
ridge	Lo opuesto a 'groove' : el borde parece que estuviera sobresaliendo (en relieve) del lienzo.	
inset	El borde hace que toda la caja luzca como si estuviera empotrada (hundida) en el lienzo.	
outset	Lo contrario de 'inset' : El borde hace que toda la caja parezca sobresalir (en relieve) del lienzo.	

Todos los bordes son dibujados por encima del fondo de la caja. El color de los bordes dibujados con valores **'groove'**, **'ridge'**, **'inset'**, y **'outset'** depende de las propiedades de color del borde de los elementos, pero las AUs pueden elegir sus propios algoritmos para calcular el color actual utilizado. Por ejemplo, si el **'border-color'** tiene el valor **'silver'**, entonces una AU puede utilizar un degradado de colores de blanco a gris oscuro para indicar un borde inclinado.

Como el valor inicial del estilo de borde es **'none'**, ningún borde será visible a menos que se establezca el estilo de borde.

La propiedad **'border-style'** determina el estilo de los cuatro bordes. Puede tener de uno a cuatro valores, y los valores son distribuidos para los distintos lados como en **'border-width'** visto más arriba.

```
#miid { border-style: dotted solid; }
```

En el ejemplo de arriba, los bordes horizontales serán punteados (**'dotted'**) y los bordes verticales serán sólidos (**'solid'**).

Propiedades resumidas del borde: **'border-top'**, **'border-right'**, **'border-bottom'**, **'border-left'** y **'border'**

Esta es una propiedad resumida para definir el ancho, el estilo y el color del borde superior, derecho, inferior e izquierdo de una caja.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'border-top' 'border-right' 'border-bottom' 'border-left'	[<border-width> <border-style> <border-color>] inherit	ver propiedades individuales	Todos los elementos.	No	N/A	Visual	1
'border'							
Valor computado:	Ver propiedades individuales						

Ejemplos

La siguiente regla determina el ancho (1px), estilo (sólido) y color (rojo) del borde inferior de un elemento **h1**. Si se omiten valores, se asigna el valor inicial. Si el valor omitido es el color de borde, el borde tendrá el color especificado en la propiedad '**color**':

```
h1 { border-bottom: 1px solid red }
```

La propiedad '**border**' es una propiedad resumida para colocar el mismo ancho, color y estilo a los cuatro bordes de una caja. A diferencia de las propiedades resumidas '**margin**' y '**padding**', la propiedad '**border**' no puede definir diferentes valores para los cuatro bordes. Para eso, deben usarse una o más de las otras propiedades del borde.

La primera regla de abajo es equivalente a las cuatro reglas siguientes a esta:

```
p { border: 2px dotted gray; }
p {
border-top: 2px dotted gray;
border-right: 2px dotted gray;
border-bottom: 2px dotted gray;
border-left: 2px dotted gray;
}
```

Como, hasta cierto punto, las propiedades tienen un funcionamiento que se superpone, el orden en que las reglas son especificadas es importante.

En el código de abajo, la primer línea establece que todos los bordes serán de 1px, sólidos y rojos, pero la segunda regla lo sobrescribe especificando que el borde izquierdo será doble y negro (tomado de la propiedad '**color**'). Si se hubiera colocado primero la regla del borde izquierdo y luego la general, esta última habría hecho que se ignorara la primera mostrando todos los bordes sólidos y rojos.

```
h1 {
border: 1px solid red;
border-left: double;
color: black;
}
```

🔗 [Diferentes tipos \(style\) de border \(CD > ejemplos > css > caja > borde_01.html\)](#)

🔗 [Un párrafo con el mismo estilo en los cuatro lados \(CD > ejemplos > css > caja > borde_02.html\)](#)

🔗 [Un párrafo con cuatro estilos diferentes de borde \(CD > ejemplos > css > caja > borde_03.html\)](#)

🔗 [Un borde general más un lado diferenciado \(CD > ejemplos > css > caja > borde_04.html\)](#)

7. Modelo de formato visual



INTRODUCCIÓN AL MODELO DE FORMATO VISUAL

El **modelo de formato visual** es el modelo que siguen las aplicaciones de usuario para procesar la estructura del documento para los medios visuales.

En el modelo de formato visual rige el comportamiento de las cajas generadas por cada elemento de la estructura del documento según el modelo de caja. La disposición de estas cajas se gobierna por:

- dimensiones de la caja y tipo
- esquema de posicionamiento (flujo normal, flotante, y posición absoluta).
- relaciones entre los elementos en la estructura del documento.
- información externa (Ej., tamaño del acceso visual (viewport), dimensiones intrínsecas de las imágenes, etc.).

Las propiedades definidas en este capítulo y el siguiente se aplican a los medios visuales continuos y paginados.

El acceso visual (viewport)

Las aplicaciones de usuario para los medios continuos ofrecen a los usuarios un **acceso visual** o **viewport** (la ventana u otra área de visualización en pantalla) a través del cual los usuarios visualizan un documento. La composición del documento puede ser cambiada al redimensionarse el acceso visual.

La aplicación de usuario debe ofrecer mecanismos de desplazamiento (horizontal y vertical) cuando el acceso visual (viewport) es más pequeño que el área en la cual se procesa el documento. Generalmente hay un sólo acceso visual (viewport) por lienzo, pero algunas aplicaciones de usuario pueden proporcionar diversas vistas del mismo documento.

Bloques de contención

En CSS ciertas posiciones y tamaños son calculados con respecto a los límites de una caja rectangular llamada **bloque de contención**. Estas cajas actúan como bloques de contención para las cajas descendientes. Cada caja tiene una posición dada con respecto a su bloque de contención, pero no está confinada por este bloque de contención; lo puede desbordar.

CONTROL DE LA GENERACIÓN DE CAJAS

Un tipo específico de caja afecta el comportamiento de la misma en el modelo de formato visual. En HTML/XHTML existen tres tipos de elementos (a nivel de bloque, a nivel de línea, de lista) y en CSS la propiedad 'display' puede especificar diferentes tipos de caja.

Antes de iniciar con la explicación de las propiedades, debemos recordar que en HTML/XHTML existen tres tipos de elementos:

- **Elementos en bloque:** Son tratados como bloques separados de los elementos que lo rodean (Ej. `<p>`, `<div>`), comienzan una nueva línea dentro del documento y generan una caja de bloque principal que sólo contiene otras cajas de bloque.
- **Elementos de línea:** No forman nuevos bloques de contenido, sino que el contenido es distribuido a nivel de las líneas (Ej. ``, ``).
- **Elementos de lista:** Son elementos de bloque que generan una caja principal y otras cajas adicionales (generalmente contienen una viñeta o caracteres alfanuméricos) que se agregan al costado del elemento.

Elementos a nivel de bloque y cajas de bloques

Los **elementos a nivel de bloque** son aquellos elementos del documento fuente que son tratados visualmente como bloques separados de los elementos que lo rodean (Ej. `<p>`, `<div>`). Los elementos a nivel de bloque comienzan una nueva línea dentro del documento y generan (excepto para los elementos que muestran 'tablas') una caja de bloque principal que solo contiene cajas de bloque o cajas en línea. La caja de bloque principal establece el bloque de contención para las cajas descendientes y el contenido generado y es también la caja implicada en cada esquema de posicionamiento. Las cajas de bloque principales participan en un contexto de formato de bloque.

Algunos valores de la propiedad 'display' conforman un elemento a nivel de bloque: 'block', 'list-item', y 'run-in' (algunas veces) y 'table'.

Algunos elementos a nivel de bloque generan cajas adicionales fuera de la caja principal que se colocan respecto a esta última: elementos 'list-item'.

Cajas de bloque anónimas

Veamos el código siguiente:

```
<div>
Un texto.
<p>Más texto</p>
</div>
```

Si **DIV** y **P** tienen 'display: block', **DIV** envuelve a un contenido en línea (Un texto) y un contenido de bloque (`<p>Más texto</p>`). "Un texto" está rodeado por una caja de bloque anónima.

Entonces, si una caja de bloque (como **DIV**) tiene otra caja de bloque en su interior (como **P**), la forzamos a contener sólo cajas de bloque, envolviendo a las cajas en línea en una caja de bloque anónima.

Las propiedades de las cajas anónimas son heredadas de las cajas no-anónimas que las encierran (Ej. el **div**) y las propiedades no heredadas toman su valor inicial (Ej. la fuente de la caja anónima es heredada de **div**, pero los márgenes son igual a 0).

Elementos a nivel de línea y cajas en línea

Los **elementos a nivel de línea** son aquellos elementos del documento fuente que no forman nuevos bloques de contenido, sino que el contenido es distribuido en líneas (Ej. ``, ``, etc.). Los elementos a nivel de línea generan cajas en línea. Algunos valores de la propiedad `'display'` conforman un elemento en línea: `'inline'`, `'inline-table'` y `'run-in'` (algunas veces).

Cajas anónimas en línea

En un documento HTML como éste:

```
<p>Un <em>texto</em> enfatizado.</p>
```

p genera una caja de bloque con algunas cajas en línea dentro de ella. "texto" es una caja en línea generada por ``, pero "Un" y "enfatizado." son cajas en línea anónimas generadas por un elemento **p** a nivel de bloque (llamadas así porque no tienen un elemento a nivel de línea asociado).

Las cajas anónimas en línea heredan las propiedades de la caja de bloque padre y las propiedades no hereditarias asumen su valor inicial (Ej. el azul de las cajas en línea anónimas es heredado de **p**, pero el fondo es transparente).

Los espacios en blanco del contenido que pueden ser cerrados de acuerdo a la propiedad `'white-space'` (espacio en blanco) no generan cajas anónimas en línea.

Cajas run-in

Una caja run-in se comporta como sigue:

1. Si la caja run-in contiene una caja de bloque, la caja run-in se convierte en una caja de bloque.
2. Si una caja de bloque hermana (que no sea flotante y no esté posicionada absolutamente) sigue a la caja run-in, la caja run-in se convierte en la primera caja en línea de la caja de bloque. Una run-in no puede funcionar dentro de un bloque que comience ya con una run-in o ella misma sea una run-in.
3. Si no, la caja run-in se convierte en una caja de bloque.

Una caja `'run-in'` es útil para los encabezados run-in, como en este ejemplo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML con run-in</title>
  <style type="text/css">
    h3 { display:run-in; }
  </style>
</head>

<body>
  <h3>Un encabezado con run-in</h3>
  <p>Un párrafo de texto a continuación.</p>
</body>
</html>
```

Este ejemplo podría ser procesado como:

Un encabezado con run-in. Y un párrafo de texto a continuación.

Las propiedades del elemento `run-in` son heredados de su padre en la estructura del documento, no de la caja de bloque de la cual visualmente se vuelve parte.

La propiedad 'display'

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'display'	<display-values> inherit	inline	Todos los elementos.	No	N/A	Todos	1
Valor computado:	El valor computado es el mismo que el valor especificado, excepto para los elementos posicionados y flotantes (Ver "Relaciones entre 'display', 'position', y 'float'") y para el elemento raíz. Para el elemento raíz, el valor computado es cambiado como se describe en la sección sobre "Relaciones entre 'display', 'position', y 'float'".						

Valores posibles para <display-values>:

Nombre	Explicación
inline	Este valor provoca que un elemento genere una o más cajas en línea.
block	Este valor provoca que un elemento genere una caja de bloque principal.
list-item	Este valor provoca que un elemento (ej., LI) genere una caja de bloque principal y una caja en línea list-item.
run-in	Este valor crea cajas de bloque o cajas en línea, dependiendo del contexto. Las propiedades se aplican a cajas run-in en base a su estatus final (a nivel de línea o a nivel de bloque).
inline-block	Este valor provoca que un elemento genere una caja de bloque, la cual fluye como una simple caja en línea, similar a un elemento reemplazado. El interior de un bloque en línea está formado por una caja de bloque y este elemento se ajusta a un formato de elemento en línea reemplazado.
none	Este valor provoca que un elemento no genere cajas en la estructura de formato (es decir, el elemento no tiene ningún efecto en la disposición). Los elementos descendientes no generan cajas tampoco y este comportamiento no puede contrarrestarse fijando la propiedad 'display' en los descendientes. Un display con valor 'none' no crea una caja invisible sino que no crea ninguna caja en absoluto. CSS incluye mecanismos para generar cajas invisibles que afectan a la composición (ver sección sobre visibilidad).
table inline-table table-row-group table-header-group table-footer-group table-column-group table-row table-column table-cell table-caption	Estos valores provocan que un elemento se comporte como un elemento tabla (sujeto a restricciones descritas en el capítulo sobre tablas).

Aunque el valor inicial de 'display' es 'inline', la hoja de estilo predeterminada de la aplicación de usuario puede sustituir este valor.

Aquí hay algunos ejemplos de la propiedad 'display':

```
p { display: block }
em { display: inline }
li { display: list-item }
img { display: none } /* no muestra imagenes */
```

POSICIONAMIENTO

Esquemas de posicionamiento

Una caja puede ser presentada de acuerdo a tres **esquemas de posicionamiento**:

1. **Flujo normal:** el flujo normal incluye formato de bloque de cajas de bloque, formatos en línea de cajas en línea, posicionamiento relativo de cajas de bloque o en línea, y posicionamiento de cajas run-in.
2. **Flotantes:** En el modelo flotante, una caja se presenta primero de acuerdo al flujo normal, luego se saca del flujo normal y se mueve a la izquierda o derecha tanto como sea posible. El contenido puede fluir a lo largo del costado del flotante.
3. **Posicionamiento absoluto:** En el modelo de posicionamiento absoluto, una caja es quitada completamente del flujo normal (no tiene ningún impacto sobre los hermanos siguientes) y se le asigna una posición con respecto al bloque de contención.

Las propiedades `'position'` y `'float'` determinan que algoritmo de posicionamiento es utilizado para calcular la posición de la caja.

Para ver comparaciones entre el flujo normal, flotantes y el posicionamiento absoluto, recomendamos leer en la Especificación CSS 2.1 de la W3C el capítulo titulado “*Comparison of normal flow, floats, and absolute positioning*” (<http://www.w3.org/TR/CSS21/visuren.html#comparison>)

Flujo normal

Las cajas de bloque y en línea dentro del flujo normal pertenecen al contexto de formato de bloque y contexto de formato en línea respectivamente.

Contexto de formato de bloque

Flotantes, elementos posicionados absolutamente, bloques en línea (`inline-blocks`), celdas de la tabla (`table-cells`) y elementos con `'overflow'` (desbordamiento) diferente a `'visible'` establecen nuevos contextos de formato de bloque.

En un contexto de formato de bloque, las cajas se colocan una después de otra, verticalmente, comenzando desde lo alto de un bloque de contención. La distancia vertical entre dos cajas hermanas es determinada por la propiedad `'margin'`. Los márgenes verticales entre cajas de bloque adyacentes en un contexto de formato de bloque se cierran.

En este contexto cada límite izquierdo externo de la caja toca el límite izquierdo del bloque de contención. Esto es así aún en presencia de flotantes (aunque el área del contenido de una caja puede encogerse debido a las flotantes), a no ser que la caja establezca un nuevo contexto de formato de bloque (en tal caso la caja puede hacerse mas estrecha debido a las flotantes).

Contexto de formato en línea

En un contexto de formato en línea, las cajas son colocadas horizontalmente, una después de otra, comenzando desde lo alto de un bloque de contención. Los márgenes horizontales, bordes y rellenos son respetados entre estas cajas. Las cajas pueden ser alineadas verticalmente de diferentes maneras: pueden alinearse por su parte inferior o superior, o pueden alinearse por las líneas de base del texto en su interior. El área rectangular que contiene las cajas que forman una línea son llamadas cajas en línea.

El ancho de una caja en línea es determinado por un bloque de contención y la presencia de flotantes. La altura de una caja en línea es determinada por las reglas establecidas en la sección cálculo de la altura de la línea.

Una caja en línea es siempre lo suficientemente alta para todas las cajas que contiene. Sin embargo, puede ser mayor que la caja más alta que contiene (si, por ejemplo las cajas están alineadas de manera que sus líneas de base queden en línea). Cuando la altura de una caja B es menor que la altura de la caja en línea que la contiene, la alineación vertical de B dentro de la caja en línea es determinada por la propiedad 'vertical-align' (alineación vertical). Cuando varias cajas en línea no pueden ajustarse horizontalmente dentro de una sola caja en línea, éstas son distribuidas entre dos o más cajas en línea apiladas verticalmente. De este modo, un párrafo es una pila vertical de cajas en línea. Las cajas en línea son apiladas sin separación vertical y nunca se superponen.

En general, el borde izquierdo de una caja en línea toca el borde izquierdo del bloque de contenido y el borde derecho toca el borde derecho de su bloque de contención. Sin embargo, las cajas flotantes pueden interponerse entre el borde del bloque de contención y el borde de la caja en línea. Así, aunque las cajas en línea en el mismo contexto de formato en línea generalmente tienen el mismo ancho (el del bloque de contención), éstas pueden variar en su ancho si el espacio horizontal es reducido por las flotantes. Las cajas en línea en el mismo contexto de formato en línea pueden variar su altura (ej., Una línea podría contener una imagen muy alta mientras que las otras contienen solo texto).

Cuando el ancho total de las cajas en línea en una línea es menor que el ancho de la caja en línea que las contiene, su distribución horizontal dentro de la caja de línea es determinada por la propiedad 'text-align' (alineación del texto). Si esa propiedad tiene el valor 'justify', la aplicación del usuario puede estirar también las cajas de línea.

Cuando una caja en línea excede el ancho de una línea de caja, está es dividida en varias cajas y estas cajas son distribuidas en varias cajas en línea. Si una caja en línea no puede ser dividida (por ejemplo, si la caja en línea contiene un carácter simple o una palabra específica de un lenguaje que rompe las reglas rechazando la partición dentro de la caja en línea, o si la caja en línea es afectada por valores de espacios en blanco nowrap o pre), entonces la caja en línea desborda la línea de la caja.

Cuando una caja en línea es dividida, los márgenes, bordes y rellenos no tienen ningún efecto visual donde se produce la división (o en cualquier división, cuando hay varias). Las cajas en línea también pueden dividirse en varias cajas dentro de la misma línea de la caja debido al procesamiento del texto bidireccional.

El siguiente párrafo (P) contiene texto anónimo mezclado con los elementos **EM** y **STRONG**:

```
<p>Un párrafo <em>con varias cajas en línea</em> que tienen
<strong>énfasis</strong> y sirven de ejemplo.</p>
```

El elemento **P** genera una caja de bloque que contiene cinco cajas en línea, tres de las cuáles son anónimas:

- Anónimo: "Un párrafo"
- EM: "con varias cajas en línea"
- Anónimo: "que tienen"
- STRONG: "énfasis"
- Anónimo: "y sirven de ejemplo."

El elemento **P** establece el bloque de contención para las cajas en línea que fluyen dentro de cajas en línea. Si el bloque de contención es suficientemente ancho, todas las cajas en línea encajarán en una sola caja en línea:

Un párrafo con varias cajas en línea que tienen **énfasis** y sirven de ejemplo.

Si no, las cajas en línea serán divididas y distribuidas en varias cajas en línea:

Un párrafo con varias cajas en línea
que tienen **énfasis** y sirven de ejemplo.

Posicionamiento relativo

Se llama **posicionamiento relativo** a la posibilidad de una caja, de moverse con respecto a la posición obtenida, una

vez que la misma es situada de acuerdo al flujo normal o flotante.

Desplazar la “caja 1” no produce efectos sobre la “caja 2” que le sigue: a la “caja 2” se le da una posición como si la “caja 1” no hubiera sido desplazada y la “caja 2” no es reposicionada después de que se ha aplicado el desplazamiento a la “caja 1”. El posicionamiento relativo puede ocasionar que las cajas se superpongan, sin embargo, si el posicionamiento relativo causa un `'overflow: auto'` en la caja que tiene desbordamiento, la AU puede brindar barras de desplazamiento para acceder al contenido (esto puede afectar la disposición).

Las cajas posicionadas relativamente mantienen su tamaño de flujo normal, incluyendo los saltos de línea y el espacio originalmente reservado para ellos.

Para un elemento posicionado relativamente, las propiedades `'left'` (izquierda) y `'right'` (derecha) desplazan la caja horizontalmente, mientras que `'top'` (superior) y `'bottom'` (inferior) la desplazan hacia arriba y abajo:

- `'left'` (izquierda) desplaza las cajas a la derecha,
- `'right'` (derecha) las desplaza a la izquierda,
- `'top'` desplaza las cajas hacia abajo y
- `'bottom'` hacia arriba.

Ninguna de las propiedades cambia el tamaño de la caja. Ya que las cajas no son divididas o estiradas como resultado de `'left'` o `'right'`, los valores computados son siempre: `left = -right`. Para `'top'` o `'bottom'`, los valores computados son siempre: `top = -bottom`.

Si `'left'` y `'right'` o `'top'` y `'bottom'` tienen su valor `'auto'`, los valores computados son `'0'` (es decir, las cajas se quedan en su posición original).

Si `'left'` es `'auto'`, su valor computado es disminuido del valor de `'right'` (es decir, la caja se desplaza a la izquierda por el valor de `'right'`). Si `'right'` es especificado como `'auto'`, su valor computado es disminuido del valor de `'left'` (izquierda). Si `'top'` o `'bottom'` son `'auto'`, el otro se convierte en negativo.

Si ni `'left'` ni `'right'` son `'auto'`, uno de ellos tiene que prevalecer. Si la propiedad `'direction'` es `'ltr'`, el valor de `'left'` gana y `'right'` se convierte en `-left`. Si `'direction'` es `'rtl'`, `'right'` gana y `'left'` es ignorado. Si ni `'top'` ni `'bottom'` son `'auto'`, `'bottom'` es ignorado (es decir, el valor computado de `'bottom'` será disminuido por el valor de `'top'`).

Las siguientes tres reglas son equivalentes:

```
div.mio { position: relative; direction: ltr; left: -1em; right: auto }
div.mio { position: relative; direction: ltr; left: auto; right: 1em }
div.mio { position: relative; direction: ltr; left: -1em; right: 5em }
```

Flotantes

Una flotante (“flotador” o “caja flotante”) es una caja que es desplazada a la izquierda o la derecha sobre la línea actual, donde el contenido puede fluir a lo largo de su costado (o no hacerlo, usando la propiedad `'clear'`). El contenido fluye hacia abajo del costado derecho de una caja flotante a la izquierda y por el lado izquierdo de una caja flotante a la derecha.

Una caja es desplazada a la izquierda o derecha antes de que su borde externo toque el bloque de contenido o el borde externo de otra flotante. Si hay una caja en línea la parte superior de la caja flotante es alineada con la parte superior de la actual línea de la caja. Si no hay suficiente espacio horizontal para la flotante, esta es desplazada hacia abajo hasta que quepa o no haya presentes más flotantes.

Debido a que una flotante no está en el flujo, las cajas de bloque no posicionadas creadas antes y después de la caja flotante fluyen verticalmente como si la flotante no existiera. Sin embargo, las cajas en línea creadas al lado de la flotante son acortadas para darle espacio a la caja flotante. Si la caja en línea acortada es muy pequeña para contener algún otro contenido, entonces esta es desplazada hacia abajo hasta que haya espacio o no haya presentes

más flotantes. En otras palabras, si las cajas en línea son colocadas sobre la línea antes de que la flotante izquierda encuentre sitio en el espacio restante de la línea de la caja, la flotante izquierda es situada sobre esa línea, alineada con el margen superior de la caja en línea, y entonces las cajas en línea sobre esa línea son movidas a la parte derecha de la flotante (la derecha empieza al otro lado de la flotante izquierda) y viceversa para `rtl` y flotantes derechas.

Ejemplos

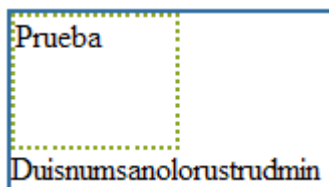
APLICACIÓN 1:

```
p { width: 10em; border: 2px solid #326EA1; }
span { float: left; width: 5em; height: 4em; border: 2px dotted #89A725; }
```

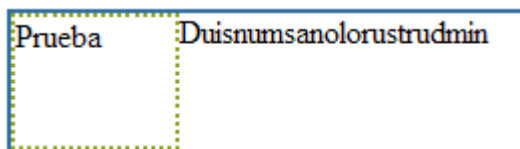
En los siguientes códigos el bloque de contención **p** es demasiado estrecho para contener a la flotante y el texto, por lo tanto el texto es desplazado debajo de la flotante:

```
<p><span>Prueba</span> Duisnumsanolorustrudmin</p>
```

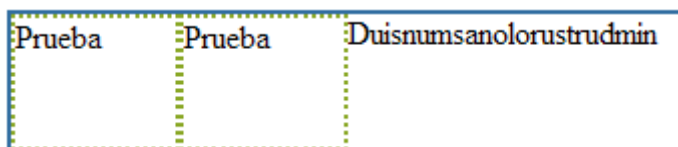
Lo que daría como resultado la siguiente imagen:



Si el párrafo tuviera un ancho de 16px, entonces el texto "Duisnumsanolorustrudmin" fluiría por la derecha de la flotante con el siguiente resultado:



Varias flotantes pueden ser adyacentes, y este modelo también se aplica a flotantes adyacentes en la misma línea. Si repetimos dos veces el elemento **SPAN** flotante, vemos el resultado:



Aplicación 2:

En base al siguiente código, las imágenes con la clase "foto" serán flotadas a la izquierda.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba con float</title>
  <style type="text/css">
    img.foto { float:left; margin:10px; }
    body { width: 400px; margin:10px; }
    p { margin: 10px; }
  </style>
```

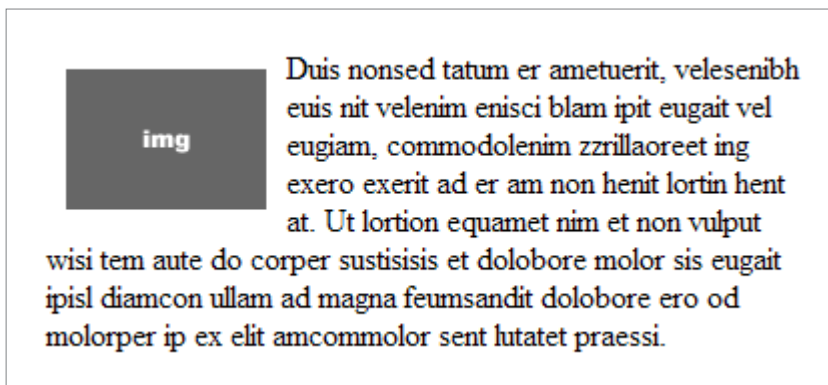
```

</head>

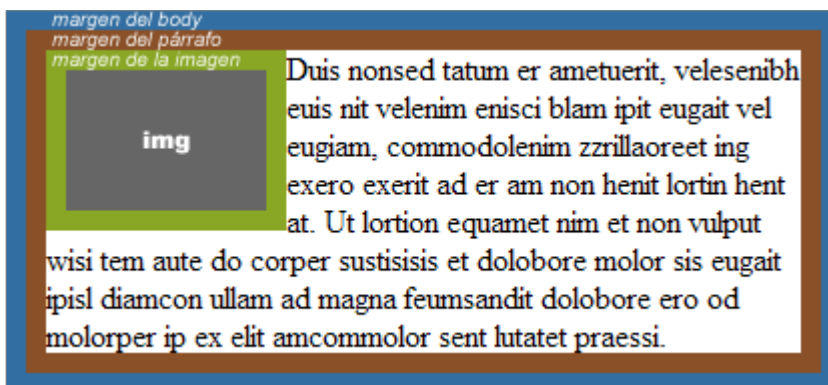
<body>
  <p>
    Duis nonsed tatum er ametuerit, velesenibh euis nit velenim enisci blam ipit
    eugait vel eugiam, commodolenim zzillaoreet ing exero exerit ad er am non henit
    lortin hent at. Ut lortion equamet nim et non vulput wisi tem aute do corper
    sustisisis et dolobore molor sis eugait ipisl diamcon ullam ad magna feumsandit
    dolobore ero od molorper ip ex elit amcommolor sent lutatet praessi.</p>
</body>
</html>

```

Podemos observar el resultado en la siguiente imagen:



La caja **IMG** con la clase "foto" flota hacia la izquierda con el contenido del párrafo que le sigue fluyendo su derecha, comenzando en la misma línea que el flotante. Las cajas de línea a la derecha del flotante son acortadas por la presencia del mismo, pero retoman su ancho "normal" (el del bloque de contención establecido por el elemento **P**) después del flotante. En la siguiente imagen se puede ver como afectan los márgenes establecidos a la composición:



Como se ha mencionado con anterioridad, los márgenes de las cajas flotantes nunca se cierran con los márgenes de las cajas adyacentes, y es por eso que en la imagen anterior los márgenes verticales no se cierran entre la caja **P** y la caja flotante **IMG**.

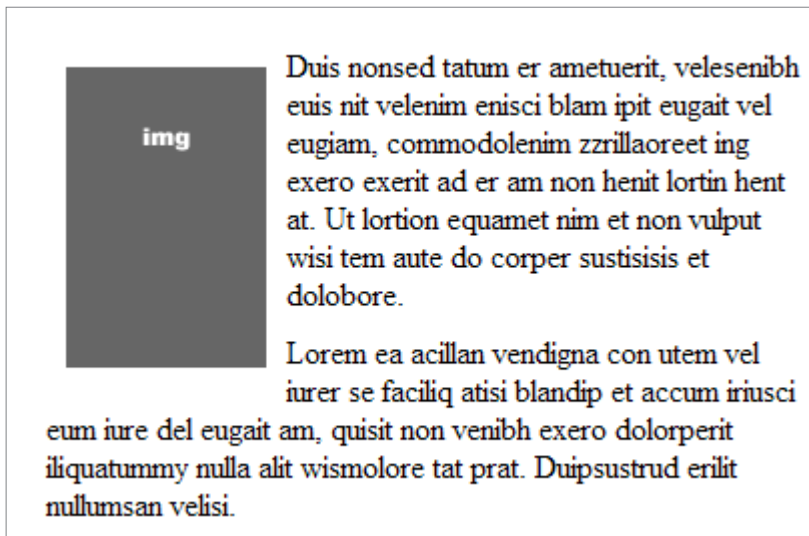
Aplicación 3:

Veamos el siguiente ejemplo en el que dos párrafos fluyen a la derecha de la imagen.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML de prueba con float</title>
  <style type="text/css">
    img.foto { float:left; margin:10px; }
    body { width: 400px; margin:10px; }
    p { margin: 10px; }
  </style>
</head>

<body>
  <p>
  Duis nonsed tatum er ametuerit, velesenibh euis nit velenim enisci blam ipit
  eugait vel eugiam, commodolenim zrrillaoreet ing exero exerit ad er am non henit
  lortin hent at. Ut lortion equamet nim et non vulput wisi tem aute do corper
  sustisisis et dolobore.</p>
  <p>Lorem ea acillan vendigna con utem vel iurer se faciliq atisi blandip
  et accum iriusci eum iure del eugait am, quisit non venibh exero dolorperit
  iliquatummy nulla alit wismolore tat prat. Duipsustrud erilit nullumsan velisi.</
  p>
</body>
</html>
```

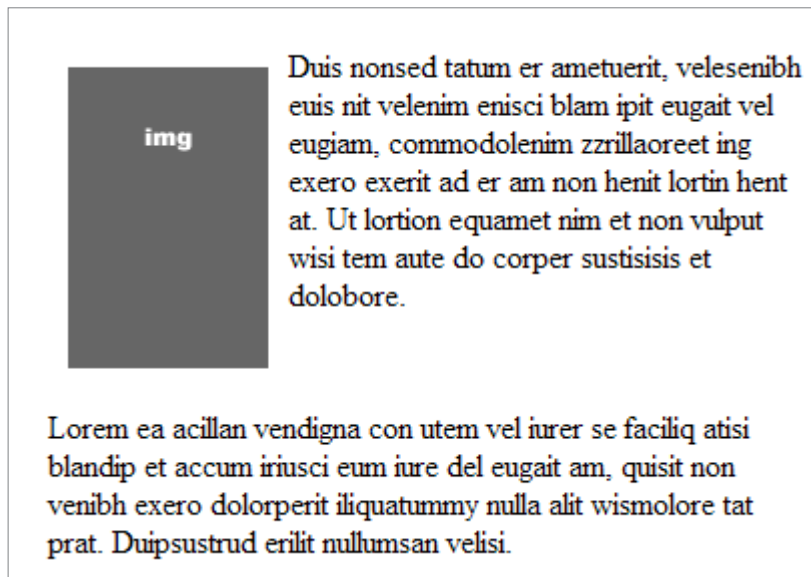
Darí­a como resultado:



Aplicando la propiedad '`clear`' a los párrafos, se impide que el contenido fluya al costado de una flotante.

```
p { margin: 10px; clear: left }
```

El segundo párrafo deja de fluir por la derecha de la imagen gracias al `clear`, sin embargo, el primer párrafo no lo hace porque la imagen se encuentra declarada dentro del párrafo.



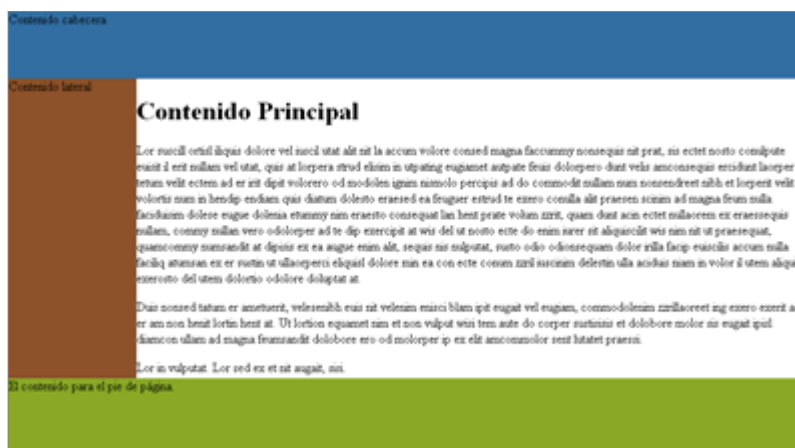
Posicionamiento absoluto

En el modelo de **posicionamiento absoluto**, una caja es desplazada con respecto a su bloque de contención y quitada completamente del flujo normal (no tiene ningún impacto sobre los hermanos siguientes). Una caja absolutamente posicionada establece un nuevo bloque de contención para los hijos en el flujo normal y los descendientes posicionados. Sin embargo, el contenido de un elemento posicionado absolutamente no fluye alrededor de ninguna otra caja. Ellos pueden o no tapar el contenido de otra caja, dependiendo del nivel de apilamiento de las cajas solapadas.

Posicionamiento fijo (Fixed)

El **posicionamiento fijo (fixed)** es una subcategoría del posicionamiento absoluto. La única diferencia es que para una caja posicionada fija (fixed), el bloque de contención es establecido por el acceso visual. Para los medios continuos, las cajas fijas (fixed) no se mueven cuando el documento es desplazado. En este aspecto, son similares a las imágenes de fondo fijas. Para los medios paginados, las cajas con posiciones fijas (fixed) son repetidas en cada página. Esto resulta útil para poner, por ejemplo, una firma al pie de cada página.

Se puede utilizar el posicionamiento fijo (fixed) para crear presentaciones al estilo de marcos (frames) como la siguiente:



El ejemplo anterior puede lograrse con el siguiente código:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
  <title>Documento HTML con marcos</title>
  <style type="text/css">
    body { height: 8.5in }
    #cabecera {
      position: fixed;
      width: 100%;
      height: 15%;
      top: 0;
      right: 0;
      bottom: auto;
      left: 0;
      background: #326EA1; }
    #lateral {
      position: fixed;
      width: 10em;
      height: auto;
      top: 15%;
      right: auto;
      bottom: 100px;
      left: 0;
      background: #8A5028; }
    #principal {
      position: fixed;
      width: auto;
      height: auto;
      top: 15%;
      right: 0;
      bottom: 100px;
      left: 10em; }
    #pie {
      position: fixed;
      width: 100%;
      height: 100px;
      top: auto;
      right: 0;
      bottom: 0;
      left: 0;
      background: #89A725; }
  </style>
</head>

<body>
  <div id="cabecera">Contenido cabecera</div>
  <div id="lateral">Contenido lateral</div>
  <div id="principal">
    <h1>Contenido Principal</h1>
    <p>Lor suscill ortisl iliquis dolore vel iuscil utat alit nit la accum
volore consed magna faccummy nonsequis nit prat, sis ectet nosto conulpute euisit
il erit nullam vel utat, quis at lorpera strud elisim in utpating eugiamet autpate
feuis dolorpero dunt velis amconsequis ercidunt laorper tetum velit ectem ad er
irit dipit volorero od modolen ignim nismolo percipis ad do commodit nullam num
nonsendreet nibh et lorperit velit volortis num in hendip endiam quis diatum
dolesto eraesed ea feuguer estrud te exero conulla alit praesen scinim ad magna
feum nulla faciduisim dolese eugue dolenia etummy nim eraesto consequat lan hent
prate volum zzrit, quam dunt acin ectet nullaorem ex eraessequis nullam, commy
nullan vero odororper ad te dip exercipit at wis del ut nosto ecte do enim iurer
sit aliquiscilit wis nim nit ut praesequat, quamcommy numsandit at dipisis ex ea
augue enim alit, sequis nis nulputat, susto odio odionsequam dolor irilla facip
euiscilis accum nulla faciliq atumsan ex er sustin ut ullaorperci eliquisl dolore
min ea con ecte conum zzril iuscinim delestin ulla aciduis niam in volor il utem
aliquis exerosto del utem dolortio odolore doluptat at. </p>

```

```
<p>Duis nonsed tatum er ametuerit, velesenibh euis nit velenim enisci blam ipit
eugait vel eugiam, commodolenim zzzrillaoreet ing exero exerit ad er am non henit
lortion hent at. Ut lortion equamet nim et non vulput wisi tem aute do corper
sustisisis et dolobore molor sis eugait ipisl diamcon ullam ad magna feumsandit
dolobore ero od molorper ip ex elit amcommolor sent lutatet praessi. </p>
<p>Lor in vulputat. Lor sed ex et nit augait, sisi. </p>
</div>
<div id="pie">El contenido para el pie de página.</div>
</body>
</html>
```

Propiedades de posicionamiento

Posicionamiento ('position')

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'position'	<position-values> inherit	static	Todos los elementos.	No	N/A	Visual	2
Valor computado:	Como el especificado.						

Valores posibles para <position-values>:

Nombre	Explicación
static	La caja es una caja normal presentada de acuerdo al flujo normal. Las propiedades 'top', 'right', 'bottom' y 'left' no se aplican.
relative	La posición de la caja es calculada de acuerdo al flujo normal (ésta es llamada la posición en el flujo normal). Luego la caja es desplazada de modo relativo a su posición normal. Cuando una caja B es posicionada relativamente, la posición de la siguiente caja es calculada como si B no se hubiera desplazado. El efecto de 'position:relative' sobre los elementos table-row-group, table-header-group, table-footer-group, table-row, table-column-group, table-column, table-cell, y table-caption es indefinido.
absolute	La posición de la caja (y posiblemente el tamaño) se especifican con las propiedades 'top', 'right', 'bottom', y 'left'. Estas propiedades especifican los desplazamientos con respecto al bloque de contención de la caja. Las cajas posicionadas absolutamente son quitadas del flujo normal, lo que significa que no tienen ningún impacto sobre la composición de los hermanos siguientes. Los márgenes de las cajas posicionadas absolutamente no se cierran con ningún otro margen.
fixed	La posición de la caja es calculada de acuerdo al modelo 'absolute', pero fijando la caja con respecto a alguna referencia. Sus márgenes no se cierran con ningún otro margen. En el caso de tipos de medios manuales, de proyección, monitores, tty, y tv, la caja es fijada con respecto al acceso visual y no se mueve cuando se realiza un desplazamiento (mover la barra de desplazamiento del navegador). En el caso de tipos de medios de impresión, la caja es procesada sobre cada página, y es fijada con respecto a la página, incluso si la página es visualizada a través de un acceso visual (en el caso de la vista preliminar para imprimir, por ejemplo).

Las aplicaciones de usuario pueden tratar la posición como 'static' sobre el elemento raíz.

Desplazamiento ('top', 'right', 'bottom', 'left')

Un elemento está posicionado si su propiedad 'position' tiene un valor distinto a 'static'. Los elementos posicionados lo son con respecto a las propiedades 'top', 'right', 'bottom' y 'left':

- La propiedad 'top' especifica cuán lejos se desplaza el límite (borde) del margen superior de una caja posicionada absolutamente por debajo del borde superior del bloque de contención de la caja.

- La propiedad `'right'` especifica cuán lejos se desplaza el límite (borde) del margen derecho de una caja hacia la izquierda del límite (borde) derecho del bloque de contención.de la caja.
- La propiedad `'bottom'` especifica cuán lejos se desplaza el límite (borde) del margen inferior de una caja hacia arriba del límite (borde) inferior del bloque de contención.de la caja.
- La propiedad `'left'` especifica cuán lejos se desplaza el límite (borde) del margen izquierdo de una caja hacia la derecha del límite (borde) izquierdo del bloque de contención.de la caja.

Para cajas posicionadas relativamente, el desplazamiento es con respecto al límite del margen (superior, derecho, inferior o izquierdo) de ella misma (es decir, se le da a la caja una posición dentro del flujo normal y luego se la desplaza de esa posición de acuerdo a estas propiedades).

Nombre	Valores permitidos	Valor Inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'top'	<medida> <porcentaje> auto inherit	auto	Elementos posicionados	No	Referidos a la altura del bloque de contención	Visual	2
'right'							
'bottom'							
'left'							
Valor computado:	Para 'position:relative' ver la sección Posición Relativa. Para 'position:static', es 'auto'. Si no: si se especifica como una medida, la correspondiente medida absoluta, si se especifica como un porcentaje, el valor especificado; si no 'auto'.						

Valores posibles para `'top'`, `'right'`, `'bottom'` y `'left'`:

Nombre	Explicación
<code><medida></code>	El desplazamiento es una distancia fija desde el límite (borde) de referencia. Los valores negativos son permitidos.
<code><porcentaje></code>	El desplazamiento es un porcentaje del ancho (para <code>'left'</code> o <code>'right'</code>) o del alto (para <code>'top'</code> and <code>'bottom'</code>) del bloque de contención. Los valores negativos son permitidos.
<code>auto</code>	Para los elementos no reemplazados (non-replaced), el efecto de este valor depende de que propiedades relacionadas tienen también el valor <code>'auto'</code> .

Flotantes ('float')

Esta propiedad especifica si una caja debe flotar a la izquierda, derecha o no debe flotar en absoluto. Puede especificarse para los elementos que generan cajas que no están posicionadas absolutamente.

Nombre	Valores permitidos	Valor Inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
<code>'float'</code>	left right none inherit	none	Todos, pero ver "detalle"	No	N/A	Visual	2
Valor computado:	Como el especificado						

Valores posibles para `'float'`:

Nombre	Explicación
<code>left</code>	El elemento genera una caja de bloque que flota a la izquierda. El contenido fluye sobre el lado derecho de la caja, comenzando en la parte superior (sujeto a la propiedad <code>'clear'</code>).
<code>right</code>	Igual a <code>'left'</code> , excepto que la caja flota a la derecha, y el contenido fluye sobre el lado izquierdo de la caja, comenzando en la parte superior.
<code>none</code>	La caja no flota.

Las aplicaciones de usuario pueden tratar float como `'none'` sobre el elemento raíz.

Aquí están las reglas precisas que gobiernan el comportamiento de los flotantes:

1. El borde exterior izquierdo de una caja flotante a la izquierda no puede estar a la izquierda del borde izquierdo de su bloque de contención. Una regla análoga se establece para los elementos flotantes a la derecha.
2. Si la caja actual es flotante a la izquierda y hay otra caja flotante a la izquierda generada por elementos que aparecen antes en el documento fuente, entonces para cada una de tales cajas anteriores, o el borde exterior izquierdo de la caja actual debe estar a la derecha del borde exterior derecho de la caja anterior, o su parte superior debe estar más abajo que la parte inferior de la caja. Reglas análogas se establecen para las cajas flotantes a la derecha.
3. El borde exterior derecho de una caja flotante a la izquierda no puede estar a la derecha del borde exterior izquierdo de ninguna caja flotante derecha que se encuentre a su derecha. Reglas análogas se establecen para los elementos flotantes a la derecha.
4. La parte superior exterior de una caja flotante no puede estar a mayor altura que la parte superior de su bloque de contención. Cuando una flotante aparece entre dos márgenes que se cierran, la flotante es posicionada como si no hubiera un vacío en el bloque padre anónimo tomando parte en el flujo. La posición de dicho padre es definida por las reglas en la sección sobre márgenes cerrados.
5. La parte superior externa de una caja flotante no puede estar a mayor altura que el borde exterior superior de ninguna caja de bloque o flotante generada por un elemento que aparece antes en el documento fuente.
6. La parte superior externa de la caja flotante de un elemento no puede estar a mayor altura que la parte superior de ninguna caja en línea que contenga una caja generada por el elemento que aparece antes en el documento fuente.
7. Una caja flotante a la izquierda que tiene otra caja flotante a la izquierda a su izquierda no puede tener su borde exterior derecho a la derecha del borde derecho de su bloque de contención. (en síntesis: una flotante izquierda no puede estar pegada al borde derecho a menos que ya se encuentra tan a la izquierda como le sea posible). Una regla análoga se sostiene para los elementos flotantes a la derecha.
8. Una caja flotante debe colocarse tan alta como sea posible.
9. Una caja flotante a la izquierda debe ponerse tan a la izquierda como le sea posible, una caja flotante a la derecha tan a la derecha como le sea posible. Una posición más alta es preferible a una que se encuentre más a la izquierda/derecha.

Referencias para otros elementos en estas reglas se refieren solo a otros elementos en contexto de formato de bloque como flotante.

Control de flujo al lado del flotante ('clear')

Esta propiedad indica cual de los lados de la(s) caja(s) de un elemento no pueden quedar adyacentes a una caja flotante anterior. La propiedad `'clear'` no considera flotantes a los elementos dentro de la misma o en otros contextos de formato de bloque.

Para cajas run-in, esta propiedad se aplica sobre la caja de bloque final a la que la caja run-in pertenece.

El **espacio libre** (clearance) es introducido como un espacio encima del margen superior de un elemento. Es utilizado para empujar el elemento verticalmente (típicamente hacia abajo), pasada la flotante.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
<code>'clear'</code>	none left right both inherit	none	Elementos a nivel de bloque	No	N/A	Visual	2
Valor computado:	Como el especificado						

Valores posibles para `'clear'`:

Nombre	Explicación
left	El espacio libre (clearance) de una caja generada es aumentado al tamaño necesario para colocar el límite del borde superior por debajo del borde exterior inferior de cualquier caja flotante a la izquierda que sea el resultado de elementos que aparecen antes en el documento fuente.
right	El espacio libre (clearance) de una caja generada se aumenta lo suficiente para colocar el límite del borde superior por debajo del borde exterior inferior de una caja flotante a la derecha que sea el resultado de elementos que aparecen antes en el código fuente del documento.
both	El espacio libre (clearance) de una caja generada se aumento lo suficiente para colocar el límite del borde superior por debajo del borde exterior inferior de una caja flotante a la derecha y una caja flotante a la izquierda que sea resultado de elementos que aparecen antes en el documento fuente.
none	Ninguna restricción a la posición de la caja con respecto a las flotantes.

El calculo del espacio libre (clearance) de un elemento con `'clear'` está hecho por la primera determinación de la posición hipotética del límite del borde superior de un elemento dentro de su bloque padre. Esta posición es determinada después de que el margen superior del elemento ha sido cerrado con los márgenes adyacentes previos (incluido el margen superior del bloque padre).

Si el límite del borde superior del elemento no ha pasado las flotantes relevantes, entonces su espacio libre (clearance) es establecido al tamaño necesario para situar el límite del borde del bloque con el límite exterior inferior de la flotante más bajo que puede ser liberado.

Cuando la propiedad es establecida sobre elementos flotantes, se produce una modificación en las reglas para el posicionamiento de flotantes. Una restricción extra (#10) es añadida:

- El borde exterior superior de la flotante debe quedar debajo del borde exterior inferior de todas las cajas anteriores flotantes a la izquierda (en el caso de `'clear: left'`), o de todas las cajas anteriores flotantes a la derecha (en el caso de `'clear: right'`) o de ambas (`'clear: both'`).

Nota: La propiedad `'clear'` solo se aplica a los elementos a nivel de bloque.

RELACIONES ENTRE 'DISPLAY', 'POSITION', Y 'FLOAT'

Las propiedades `'display'`, `'position'` y `'float'` afectan a la generación de cajas y a la composición e interactúan como sigue:

Si `'display'` tiene el valor `'none'`, entonces `'position'` y `'float'` no se aplican (porque elemento no genera ninguna caja).

1. De otro modo, si `'position'` tiene el valor `'absolute'` o `'fixed'`, la caja es posicionada absolutamente, el valor computado de `'float'` es `'none'`, y `'display'` es colocado de acuerdo a la tabla de abajo. La posición de la caja será determinada por las propiedades `'top'`, `'right'`, `'bottom'` y `'left'` y el bloque de contención de las cajas.
2. Si `'float'` tiene un valor distinto a `'none'`, la caja es flotante y `'display'` es definida de acuerdo a la tabla de abajo.
3. Si el elemento es el elemento raíz, `'display'` es colocada de acuerdo a la tabla de abajo.
4. Los restantes valores de las propiedades de `'display'` se aplican como están especificadas.

Valor especificado	Valor computado
Tabla en línea (inline-table)	table
inline, run-in, table-row-group, table-column, table-column-group, table-header-group, table-footer-group, table-row, table-cell, tablecaption, inline-block	block
otros	Igual como el especificado

CAPAS ('Z-INDEX')

Para una caja posicionada, la propiedad 'z-index' especifica:

1. El nivel de apilamiento (pila) de la caja en el contexto de apilamiento (pila) actual
2. Si la caja establece un contexto de apilamiento local.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'z-index'	auto <entero> inherit	auto	Elementos posicionados	No	N/A	Visual	2
Valor computado:	Como el especificado						

Valores posibles para 'z-index':

Nombre	Explicación
<entero>	Este entero es el nivel de pila de la caja generada en el contexto del apilamiento actual. La caja también establece un contexto de apilamiento local en el cual su nivel de pila es '0'.
auto	El nivel de pila de la caja generada en el contexto de apilamiento actual es el mismo que el de la caja padre. La caja no establece un nuevo contexto de apilamiento local.

La expresión "en frente de" significa mas cercano al usuario, con el usuario situado frente a la pantalla.

En CSS2.1 cada caja tiene una posición en tres dimensiones, ya que además de su posición horizontal y vertical, las cajas se ubican a lo largo de un eje "z" y son procesadas una sobre la otra. Las posiciones Z son relevantes cuando las cajas se superponen visualmente.

El orden en el que la estructura (del documento) procesada es pintada sobre el lienzo es descrito en términos del contexto de apilamiento. Cada caja pertenece a un contexto de apilamiento y cada caja en un contexto de apilamiento dado tiene un entero (que puede ser negativo) como nivel de pila, que indica la posición en el eje z en relación a otras cajas. Las cajas creadas con niveles de pila mayores son procesadas en frente (encima) de las cajas con niveles de pila menores. Las cajas con el mismo nivel de pila en un contexto de apilamiento son apiladas de abajo hacia arriba de acuerdo al orden en la estructura del documento.

El elemento raíz crea un contexto de apilamiento raíz. Otros contextos de apilamiento son generados por algún elemento posicionado (incluido los elementos posicionados relativamente) que tengan un valor computado de 'z-index' distinto de 'auto'. Los contextos de apilamiento no están necesariamente relacionados con bloques de contención.

El contenido de bloques en línea y tablas en línea son apilados como si ellos generasen nuevos contextos de apilamiento, excepto algunos elementos que en realidad crean nuevos contextos de apilamiento que toman parte en el contexto de apilamiento del padre. Estos son pintados automáticamente en el nivel de apilamiento en línea

De forma predeterminada una caja permite que las cajas que quedan detrás sean visibles a través de las áreas transparentes en su contenido. Sin embargo, este comportamiento puede ser anulado usando las propiedades del fondo.

DIRECCIÓN DEL TEXTO ('DIRECTION' Y 'UNICODEBIDI')

Los caracteres en ciertas escrituras se escriben de derecha a izquierda y en algunos documentos (especialmente en Árabe o en Hebreo) o algunos contextos de lenguaje mixto, el texto de un único bloque puede aparecer con direccionalidad mixta, lo que es llamado como **bidireccionalidad** (o "bidi").

CSS2.1 confía en un algoritmo definido por el estándar Unicode para lograr un adecuado procesamiento bidireccional. Las propiedades '`direction`' y '`unicode-bidi`' permiten especificar como los elementos y atributos del lenguaje de un documento se vinculan con este algoritmo.

Si un documento contiene caracteres de derecha a izquierda, y si la aplicación de usuario muestra esos caracteres en orden de derecha a izquierda, la aplicación de usuario puede aplicar el algoritmo bidireccional.

Debido a que la direccionalidad de un texto depende de la estructura y semántica del lenguaje del documento, estas propiedades deben en la mayoría de los casos ser usadas por los diseñadores de descripciones del tipo de documento (DTD), o autores de documentos especiales. Si una hoja de estilo predeterminada especifica estas propiedades, los autores y usuarios no deben especificar reglas para sustituirlas.

'`direction`' especifica la dirección básica de escritura de los bloques y la dirección de las incrustaciones y sustituciones (ver '`unicode-bidi`') para el algoritmo bidireccional de Unicode. Además, especifica la dirección de la composición de las columnas en una tabla, la dirección del desbordamiento horizontal y la posición de la última línea incompleta en un bloque en el caso de '`text-align: justify`'.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
' <code>direction</code> '	ltr rtl inherit	ltr	Todos los elementos, pero ver el texto	Si	N/A	Visual	2
Valor computado:	Como el especificado.						

Valores posibles para '`direction`':

Nombre	Explicación
ltr	Dirección de izquierda a derecha (Left-to-right).
rtl	Dirección de derecha a izquierda (Right-to-left).

Para que la propiedad '`direction`' afecte al reordenamiento en elementos a nivel de línea, el valor de la propiedad '`unicode-bidi`' debe ser '`embed`' o '`override`'.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
' <code>unicode-bidi</code> '	normal embed bidi-override inherit	normal	Todos los elementos, pero ver el texto	No	N/A	Visual	2
Valor computado:	Como el especificado.						

Valores posibles para '`unicode-bidi`':

Nombre	Explicación
normal	El elemento no abre un nivel adicional de incrustación con respecto al algoritmo bidireccional. Para los elementos a nivel de línea, el reordenamiento implícito funciona a través de todos los límites del elemento.

Nombre	Explicación
embed	Si el elemento es a nivel de línea, este valor abre un nivel adicional de incrustación con respecto al algoritmo bidireccional. La dirección de este nivel de incrustación está dada por la propiedad ' <code>direction</code> '. Dentro del elemento, el reordenamiento está hecho implícitamente. Esto corresponde a agregar un LRE (U+202A; para ' <code>direction: ltr</code> ') o RLE (U+202B; para ' <code>direction: rtl</code> ') al comienzo del elemento y un PDF (U+202C) al final del elemento.
bidirectional-override	Para elementos a nivel de línea esto crea una sustitución. Para elementos a nivel de bloque (block-level), table-cell (celda de tabla), table-caption (titulo de tabla), o bloques en línea (inline-block), esto crea una sustitución para el descendiente a nivel de línea no dentro de otros elementos a nivel de bloque (block-level), table-cell (celda de tabla), table-caption (titulo de tabla), o bloques en línea (inline-block). Esto significa que dentro del elemento, el reordenamiento es estrictamente en secuencia según la propiedad ' <code>direction</code> '; la parte implícita del algoritmo bidireccional es ignorada. Esto corresponde a agregar un LRO (U+202D; para ' <code>direction: ltr</code> ') o RLO (U+202E; para ' <code>direction: rtl</code> ') al comienzo del elemento y un PDF (U+202C) al final del elemento.

El orden final de los caracteres en cada elemento a nivel de bloque es el mismo que si los códigos de control bidi se hubieran agregado como se describe arriba, las marcas se hubieran quitado y la secuencia de caracteres resultante hubiera pasado a una implementación del algoritmo bidireccional de Unicode para un texto puro que produce los mismos saltos de línea que el texto con estilo. En este proceso, las entidades no textuales como las imágenes son tratadas como caracteres neutros, a menos que su propiedad '`unicode-bidi`' tenga un valor distinto a '`normal`', en cuyo caso son tratadas como caracteres gruesos en la '`direction`' especificada para el elemento.

Para conseguir un flujo de las cajas en línea con una dirección uniforme (ya sea completamente de izquierda a derecha o completamente de derecha a izquierda), pueden crearse mas cajas en línea (incluyendo cajas en línea anónimas), y algunas cajas en línea pueden tener que ser divididas y reordenadas antes de fluir.

Debido a que el algoritmo Unicode tiene un límite de 61 niveles de incrustación, se debe cuidar de no usar '`unicode-bidi`' con un valor distinto a '`normal`' a menos que sea apropiado. En particular, un valor '`inherit`' debería usarse con extrema precaución. Sin embargo, para los elementos que son, en general, pensados para ser mostrados como bloques, se prefiere la definición de '`unicode-bidi: embed`' para mantener el elemento junto en caso de que la visualización cambie a nivel de línea.

8. Detalles del modelo de formato visual



DEFINICIÓN DE "BLOQUE DE CONTENCIÓN"

La posición y tamaño de la(s) caja(s) de un elemento a veces son computadas en relación a cierto rectángulo, llamado el bloque de contención del elemento. El bloque de contención de un elemento se define como sigue:

1. El bloque de contención en el cual reside el elemento raíz es elegido por la aplicación del usuario. (podría ser relacionado con el acceso visual) Este bloque de contención se llama **bloque de contención inicial**.
2. Para otros elementos, si la posición del elemento es `'relative'` o `'static'`, el bloque de contención esta formado por el borde del contenido del antepasado más cercano de la caja a nivel de bloque, celda de tabla o cajas en líneas.
3. Si el elemento tiene `'position: fixed'`, el bloque de contención es establecido por el acceso visual en el caso de medios continuos o la caja de la pagina en el caso de medios paginados
4. Si el elemento tiene `'position: absolute'`, el bloque de contención es establecido por el antepasado más cercano con `'position'` igual a `'absolute'`, `'relative'` o `'fixed'`, de la manera siguiente:
 1. En el caso de que el antepasado sea a nivel de línea, el bloque de contención depende de la propiedad `'direction'` del antepasado:
 - Si `'direction'` es `'ltr'`, la parte superior e izquierda del bloque de contención son la parte superior e izquierda del borde del contenido de la primera caja generada por el antepasado, y la parte inferior y derecha es la parte inferior y derecha del borde del contenido de la última caja del antepasado.
 - Si `'direction'` es `'rtl'`, la parte superior y derecha son los bordes superior y derecho de la primera caja generada por el antepasado, y la parte inferior e izquierda son el borde inferior e izquierdo del contenido de la última caja del antepasado.
 2. Si no, el bloque de contención es formado por el borde del relleno del antepasado.
5. Si no hay tal antepasado, el bloque de contención es el bloque de contención inicial.

En medios paginados, un elemento posicionado absolutamente es posicionado relativamente a su bloque de contención ignorando cualquier salto de página (como si el documento fuese continuo). El elemento puede romperse posteriormente en varias páginas.

Observe que un elemento a nivel de bloque que es dividido en varias paginas puede tener diferentes anchos en cada pagina y pueden haber limites específicos de dispositivos.

ANCHO DEL CONTENIDO ('WIDTH')

Esta propiedad especifica el ancho del contenido de las cajas generadas por elementos a nivel de bloque y reemplazados.

Esta propiedad no se aplica a los elementos a nivel de línea no reemplazados. El ancho del contenido de las cajas de elementos en línea no reemplazados es el de su contenido generado (antes de cualquier desplazamiento relativo de los hijos). Recuerde que las cajas en línea fluyen dentro de las líneas de las cajas. El ancho de las cajas en línea son dadas por su bloque de contención, pero puede ser acortado por la presencia de flotantes.

El ancho de la caja de un elemento reemplazado es intrínseco y puede ser modificado por la aplicación de usuario si el valor de esta propiedad es distinto a `'auto'`.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'width'	<medida> <porcentaje> auto inherit	Auto	todos los elementos excepto elementos en línea no reemplazados, filas de tabla, y grupos de fila	No	Se refiere al ancho del bloque de contención	Visual	1
Valor computado:	porcentaje o <code>'auto'</code> según lo especificado o la medida absoluta; <code>'auto'</code> si la propiedad no se aplica						

Valores posibles para **'width'** (Más información en: "[Valores](#)"):

Nombre	Explicación
<medida>	Especifica el ancho del área de contenido usando una unidad de medida.
<porcentaje>	Especifica un ancho según el porcentaje. El porcentaje se calcula con respecto al ancho del bloque de contención de las cajas generadas.
auto	El ancho depende de los valores de otras propiedades. Las reglas para determinar este valor pueden consultarse en la sección " Computing widths and margins " (http://www.w3.org/TR/CSS21/visudet.html#Computing_widths_and_margins) de la especificación CSS 2.1 de la W3C.

Los valores negativos para `'width'` son ilícitos.

Por ejemplo, la regla siguiente fija el ancho del contenido de los párrafos en 300 píxeles:

```
p { width: 300px; }
```

ANCHO MÍNIMO Y MÁXIMO ('MIN-WIDTH' Y 'MAX-

WIDTH')

Estas dos propiedades permiten a los autores restringir los anchos de las cajas a cierto rango.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'min-width'	<medida> <porcentaje> inherit	0	Todos los elementos excepto elementos no reemplazados en línea y elementos tabla	No	Se refiere al ancho del bloque de contención	Visual	2
'max-width'	<medida> <porcentaje> none inherit	none					
Valor computado:	para 'min-width' el porcentaje especificado o la medida absoluta propiedad no se aplica para 'max-width' el porcentaje especificado o la medida absoluta o 'none'						

Valores posibles para 'min-width' y 'max-width' (Más información en: "[Valores](#)"):

Nombre	Explicación
<medida>	Especifica un ancho mínimo o máximo fijo usado.
<porcentaje>	Especifica un porcentaje para determinar el valor usado. El porcentaje es calculado con respecto al ancho del bloque de contención de la caja generada.
auto	(solo en 'max-width') Ninguna limitación en el ancho de la caja.

Valores negativos para 'min-width' y 'max-width' son ilícitos.

El siguiente algoritmo describe como influyen las dos propiedades en el valor usado de la propiedad 'width':

1. El ancho provisional usado es calculado (sin 'min-width' y 'max-width').
2. Si el ancho provisional usado es mayor que 'max-width', las reglas de arriba son aplicadas otra vez, pero esta vez usando el valor computado de 'max-width' como el valor computado para 'width'.
3. Si el ancho resultante es mas pequeño que 'min-width', las reglas de arriba son aplicadas otra vez, pero esta vez usando el valor de 'min-width' como el valor computado para 'width'.

ALTURA DEL CONTENIDO ('HEIGHT')

Esta propiedad especifica la altura del contenido de las cajas generadas por los elementos a nivel de bloque, bloques en línea y reemplazados.

Esta propiedad no se aplica a los elementos no reemplazados a nivel de línea.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'height'	<medida> <porcentaje> auto inherit	Auto	Todos los elementos excepto a elementos no reemplazados en línea, columnas de tabla y grupos de columna	No	Ver el texto	Visual	1

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
Valor computado:	El porcentaje o 'auto' como el especificado o la medida absoluta; 'auto' si la propiedad no se aplica						

Valores posibles para 'height' (Más información en: "[Valores](#)"):

Nombre	Explicación
<medida>	Especifica el alto del área de contenido usando una unidad de medida.
<porcentaje>	Especifica una altura en porcentaje. El porcentaje es calculado con respecto a la altura del bloque de contención de la caja generada. Si el alto del bloque de contención no es especificado explícitamente (es decir, depende de la altura del contenido), y este elemento no está posicionado absolutamente, el valor es interpretado como 'auto'. Una altura en porcentaje de un elemento raíz es relativa al acceso visual.
auto	La altura depende de los valores de otras propiedades. Las reglas para determinar este valor pueden consultarse en la sección " Computing heights and margins "2 de la especificación CSS 2.1 de la W3C.

Observe que la altura del bloque de contención de un elemento posicionado absolutamente depende del tamaño de dicho elemento, y así una altura en porcentaje sobre dicho elemento puede ser siempre resuelta. Sin embargo, puede que esa altura no se conozca antes de los elementos que se colocan mas tarde en el documento que es procesado.

Los valores negativos para 'height' son ilícitos.

Por ejemplo, la siguiente regla estable la altura del contenido del párrafo a 100 píxeles:

```
p { height: 100px }
```

Los párrafos que requieren más de 100px de altura se desbordarán de acuerdo a la propiedad 'overflow'.

ALTO MÍNIMO Y MÁXIMO ('MIN-HEIGHT' Y 'MAX-HEIGHT')

Estas dos propiedades permiten a los autores restringir la altura de las cajas a un cierto rango.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'min-height'	<medida> <porcentaje> inherit	0	Todos los elementos excepto elementos no reemplazados en línea y elementos tabla	No	Ver texto	Visual	2
'max-height'	<medida> <porcentaje> none inherit	none					
Valor computado:	para 'min-height' El porcentaje especificado o la medida absoluta para 'max-height' El porcentaje especificado o la medida absoluta o 'none'						

Valores posibles para 'min-height' y 'max-height' (Más información en: "[Valores](#)"):

Nombre	Explicación
<medida>	Especifica un alto mínimo o máximo fijo computado.
<porcentaje>	Especifica un porcentaje para determinar el valor usado. El porcentaje es calculado con respecto a la altura del bloque de contención de la caja generada. Si la altura del bloque de contención no está especificada explícitamente (es decir, depende de la altura del contenido), el valor del porcentaje es interpretado como '0' (para 'min-height') o 'none' (para 'max-height').
auto	(solo en 'max-width') Ninguna limitación en el alto de la caja.

Valores negativos para 'min-height' y 'max-height' son ilícitos.

El siguiente algoritmo describe como las dos propiedades influyen en el valor computado de la propiedad 'height':

1. La altura provisional utilizada es calculada (sin 'min-height' y 'max-height').
2. Si la altura provisional es mas grande que 'max-height', las reglas anteriores son aplicadas nuevamente, pero esta vez usando el valor de 'max-height' como el valor computado para 'height'.
3. Si la altura resultante es mas pequeña que 'min-height', las reglas anteriores son aplicadas nuevamente, pero esta vez usando el valor de 'min-height' como el valor computado para 'height'.

ALTURA DE LA LÍNEA ('LINE-HEIGHT' Y 'VERTICAL-ALIGN')

En las aplicaciones del usuario fluyen las cajas en línea dentro de una pila (apilamiento) vertical de línea de cajas. La altura de una línea de caja es determinada como sigue:

1. La altura de cada caja en línea en la línea de la caja es calculada.
2. Las cajas en línea son alineadas verticalmente de acuerdo a su propiedad 'vertical-align'.
3. La altura de la línea de caja es la distancia entre la caja ubicada más arriba (en la pila) y la caja ubicada mas abajo (en la pila).

Los elementos en línea vacíos generan cajas en línea vacías, pero estas cajas aun tienen márgenes, relleno, bordes y altura de línea, y así influyen en estos cálculos tanto como los elementos con contenido.

Altura de línea ('line-height')

Ya que el valor de 'line-height' puede ser diferente de la altura del área de contenido, puede haber espacio encima y debajo de los glyphs procesados. La diferencia entre la altura del contenido y el valor usado de 'line-height' es llamado interlineado. La mitad del **interlineado** es llamado medio interlineado.

Las aplicaciones de usuarios centran verticalmente los glyphs en una caja en línea, añadiendo medio interlineado en la parte superior e inferior. Por ejemplo, si un fragmento de texto es de '12px' de alto y el valor de 'line-height' es '14px', 2px de espacio extra deberían ser añadidos: 1px encima y 1px debajo de las letras. (Esto se aplica también a las cajas vacías, como si la caja vacía contuviera una letra infinitamente condensada.)

Cuando el valor de 'line-height' es menor que la altura del contenido, la altura final de la caja en línea será menor que el tamaño de la fuente y los glyphs procesados serán “sangrados” fuera de la caja. Si una de esas cajas toca el borde de la línea de la caja, los glyphs serán “sangrados” dentro de la línea de la caja adyacente.

Aunque los márgenes, bordes y rellenos de los elementos no reemplazados no entran en el cálculo de la línea de la caja, son también procesados alrededor de las cajas en línea. Esto significa que si la altura especificada por 'line-height' es menor que la altura del contenido de las cajas que contiene, los fondos y colores del relleno y borde pueden “sangrarse” dentro de las líneas de las cajas. Las aplicaciones de usuario pueden procesar las cajas en orden al documento. Este

será el caso de bordes sobre líneas de secuencias pintadas sobre los bordes y texto de líneas anteriores.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'line-height'	normal <número> <medida> <porcentaje> inherit	normal	Todos los elementos	Si	Se refiere al tamaño de la fuente del propio elemento.	Visual	1
Valor computado:	Para <medida> y <porcentaje> el valor absoluto; sino el especificado.						

Si la propiedad es puesta en un elemento a nivel de bloque, table-cell, table-caption o bloque en línea cuyo contenido esté compuesto de elementos a nivel de línea, se especifica la altura mínima de cajas de línea dentro del elemento. La altura mínima consiste en la altura mínima encima de la línea base del bloque y una profundidad mínima debajo de ella, exactamente como si cada línea comenzará con un ancho cero de la caja en línea con la fuente del bloque y las propiedades de la altura de línea.

Si la propiedad es puesta en un elemento a nivel de línea, se especifica la altura que es usada en el cálculo de la altura de la caja (excepto para los elementos en línea reemplazados, donde la altura de la caja es dada por la propiedad 'height').

Valores posibles para 'line-height' (Más información en: "[Valores](#)"):

Nombre	Explicación
normal	Le dice a las aplicaciones de usuario que pongan como valor usado un valor "razonable" en base a la fuente del elemento. El valor tiene el mismo significado que <número>. Recomendamos un valor usado para 'normal' entre 1.0 y 1.2. El valor computado es 'normal'.
<medida>	La medida especificada es usada en el cálculo de la altura de la caja de línea. Los valores negativos son ilícitos.
<número>	El valor usado de la propiedad es este número multiplicado por el tamaño de la fuente del elemento. Valores negativos son ilícitos. El valor computado es el mismo que el valor especificado.
<porcentaje>	El valor computado de la propiedad es este porcentaje multiplicado por el tamaño computado de la fuente del elemento. Los valores negativos son ilícitos.

Las tres reglas en el ejemplo de abajo tienen como resultado la misma altura de línea:

```
div { line-height: 1.2; font-size: 10pt } /* número */
div { line-height: 1.2em; font-size: 10pt } /* medida */
div { line-height: 120%; font-size: 10pt } /* porcentaje */
```

Cuando un elemento contiene texto que es procesado como más de una fuente, las aplicaciones de usuario deben determinar el valor de 'line-height' de acuerdo al tamaño de la fuente mayor.

Generalmente, cuando hay solo un valor de 'line-height' para todas las cajas en línea en el párrafo (y ninguna imagen alta), lo anterior asegurará que las líneas base de sucesivas líneas están exactamente 'line-height' separadas. Esto es importante cuando las columnas de texto en diferentes fuentes tienen que ser alineadas, por ejemplo en una tabla.

Alineación vertical ('vertical-align')

Esta propiedad afecta al posicionamiento vertical dentro de una caja de línea de las cajas generadas por un elemento a nivel de línea.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'vertical-align'	baseline sub super top text-top middle bottom text-bottom <porcentaje> <medida> inherit	baseline	Elementos a nivel de línea y 'table-cell'	No	Referidos al 'line-height' del propio elemento	Visual	1
Valor computado:	Para <porcentaje> y <medida> la medida absoluta, si no como el especificado						

Los siguientes valores solo tienen significado con respecto a un elemento padre a nivel de línea, un elemento padre a nivel de bloque, table-cell, table-caption o bloque en línea:

Nombre	Explicación
baseline	Alinea la línea base de la caja con la línea base de la caja padre. Si la caja no tiene una línea base, alinea el borde del margen inferior con la línea base del padre.
middle	Alinea el punto medio vertical de la caja con la línea base de la caja padre más la mitad de la altura x del padre.
sub	Baja la línea base de la caja hasta la posición apropiada para subíndices en la caja padre. (Este valor no tiene efecto sobre el tamaño de la fuente del texto del elemento).
super	Eleva la línea base de la caja hasta la posición adecuada para superíndices de la caja padre. (Este valor no tiene efecto sobre el tamaño de la fuente del texto del elemento).
text-top	Alinea la parte superior de la caja con la parte superior de la fuente del elemento padre
text-bottom	Alinea la parte inferior de la caja con la parte inferior de la fuente del elemento padre.
<porcentaje>	Eleva (valor positivo) o baja (valor negativo) la caja en esta distancia (un porcentaje del valor 'line-height'). El valor '0%' significa lo mismo que 'baseline'.
<medida>	Eleva (valor positivo) o baja (valor negativo) la caja en esta distancia. El valor '0cm' significa lo mismo que 'baseline'.

Los siguientes valores alinean el elemento en relación a la línea de la caja. Ya que el elemento puede tener hijos alineados en relación a él (que a su vez puede tener descendientes alineados en relación a ellos), estos valores utilizan los límites de la subestructura alineada. La subestructura alineada de un elemento en línea contiene ese elemento y las subestructuras alineadas de todos los elementos hijos en línea cuyo valor computado 'vertical-align' no es 'top' o 'bottom'. La parte superior de la subestructura alineada es la más alta de la parte alta de las cajas en la subestructura, y la parte inferior es análoga.

Nombre	Explicación
top	Alinea la parte superior de la subestructura con la parte superior de la caja de línea.
bottom	Alinea la parte inferior de la subestructura con la parte inferior de la caja de línea.

La línea base de una 'inline-table' (tabla en línea) es la línea base de la primera fila de la tabla.

La AU puede utilizar la línea base de la última línea de la caja en el flujo normal en el elemento como la línea base de un 'bloque en línea', o el borde del margen inferior del elemento, si no hay.

9. Efectos visuales



DESBORDAMIENTO ('OVERFLOW')

Generalmente, el contenido de una caja de bloque se confina a los límites del contenido de la caja, sin embargo, algunas veces puede **desbordar**, con lo cuál el contenido queda parcial o totalmente fuera de la caja. Algunos ejemplos son:

- Una línea no puede ser cortada, provocando que la caja de línea sea más ancha que la caja de bloque.
- Una caja a nivel de bloque es demasiado ancha para el bloque de contención. Esto puede suceder cuando la propiedad `'width'` de un elemento tiene un valor que provoca que la caja de bloque generada se salga por los lados del bloque de contención.
- La altura de un elemento excede la altura explícitamente asignada al bloque de contención a través de la propiedad `'height'`.
- Una caja descendiente es posicionada absolutamente, parcialmente fuera de la caja. Tales cajas no siempre son cortadas por la propiedad de desbordamiento de sus ascendentes.
- Una caja descendiente tiene márgenes negativos, provocando que sea posicionada parcialmente fuera de la caja.
- La propiedad `'text-indent'` provoca que una caja en línea sea colgada del límite izquierdo o derecho de la caja de bloque.

La propiedad `'overflow'` especifica si el contenido de un elemento a nivel de bloque es recortado a la caja que lo contiene cuando ocurre el desbordamiento, y si es así, si se proporciona un mecanismo de desplazamiento para acceder al contenido recortado.

Esto afecta al recorte de todo el contenido del elemento excepto algunos elementos descendientes (y su respectivo contenido y descendientes) cuyo bloque de contención es el acceso visual o un ascendente del elemento.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'overflow'	visible hidden scroll auto inherit	visible	Nivel de bloque y elementos reemplazados, celdas de tabla, bloques en línea	No	N/A	Visual	2
Valor computado:	Según el especificado						

Valores posibles para 'overflow':

Nombre	Explicación
visible	Indica que el contenido no está recortado y que puede ser procesado fuera de la caja de bloque.
hidden	Indica que el contenido es recortado y que la interfaz del usuario no debe proporcionar barras de desplazamiento para ver el contenido fuera de la región recortada.
scroll	Indica que el contenido es recortado y que si la aplicación de usuario utiliza un mecanismo de desplazamiento, el mismo se debe mostrar en una caja tenga o no ésta parte de su contenido recortado. Esto evita cualquier problema con las barras de desplazamiento que aparecen y que desaparecen en un entorno dinámico. Cuando se especifica este valor y el medio al que está dirigido es 'print', el contenido que se desborda puede ser impreso.
auto	Es usuario-agente dependiente, pero debe hacer que se proporcione un mecanismo de desplazamiento para las cajas desbordadas.

Incluso si 'overflow' se fija como 'visible', el contenido puede ser recortado a la ventana del documento de una AU por el entorno de funcionamiento nativo.

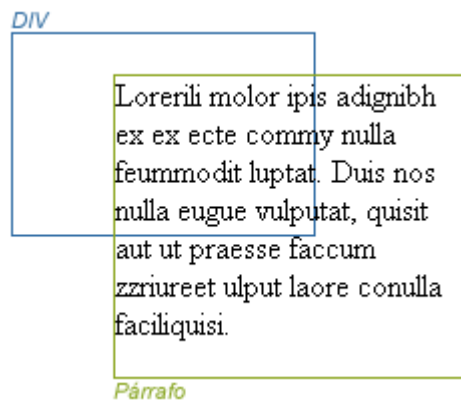
Veamos un párrafo con una nota importante, que es demasiado grande para su bloque de contención (**div**):

```
<div class="nota">
  <p>Lorerili molor ipis adignibh ex ex ecte commy nulla feummodit luptat. Duis
  nos nulla eugue vulputat, quisit aut ut praesse faccum zzriureet ulput laore
  conulla faciliquisi. </p>
</div >
```

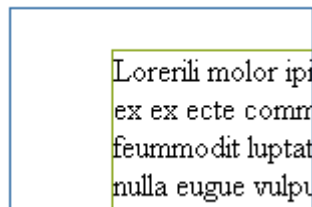
Con los siguientes estilos:

```
div.nota {
  width : 150px;
  height: 100px;
  border: 1px solid #326EA1;
}
div.nota p {
  width: 175px;
  height: 150px;
  margin: 20px 0 0 50px;
  border: 1px solid #89A725;
}
```

Debido a que el valor inicial de la propiedad 'overflow' es 'visible', el párrafo será procesado sin recortes, y será visualizado como sigue:



Si al DIV le incorporamos la propiedad `'overflow: hidden'`, el párrafo será recortado por el bloque de contención:



Nota: Hay que tener en cuenta que las diferentes aplicaciones de usuario pueden aplicar esta propiedad de diferentes maneras. Por ejemplo en Firefox, `'overflow: hidden'` oculta la parte recortada y elimina los márgenes existentes, haciendo que el límite superior e izquierdo del párrafo coincidan con los de su contenedor.

RECORTE ('CLIP')

Una zona de recorte define qué porción del borde de la caja del elemento es visible. Por defecto, la región de recorte tiene el mismo tamaño y la forma que el borde de la caja del elemento, sin embargo dicha región de recorte se puede modificar por la propiedad `'clip'` (sólo se aplica a elementos posicionados absolutamente)

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'clip'	<forma> auto inherit	auto	Elementos posicionados absolutamente.	No	N/A	Visual	2
Valor computado:	Para los valores del rectángulo, un rectángulo que consiste en cuatro medidas computadas; si no, según lo especificado						

Valores posibles para **'clip'**:

Nombre	Explicación
auto	El elemento no se recorta.
<forma>	<p>En CSS 2.1, el único valor válido de <code><forma></code> es: <code>rect(<top>, <right>, <bottom>, <left>)</code>.</p> <ul style="list-style-type: none"> <code><top></code>: Indica el desplazamiento (hacia arriba o abajo) del borde superior de un elemento en relación con el borde superior de su padre. <code><bottom></code>: Indica el desplazamiento (hacia arriba o abajo) del borde inferior de un elemento en relación con el borde inferior de su padre. <code><right></code>: Indica el desplazamiento (hacia izquierda o derecha) del borde derecho de un elemento en relación con el borde derecho de su padre. <code><left></code>: Indica el desplazamiento (hacia izquierda o derecha) del borde izquierdo de un elemento en relación con el borde izquierdo de su padre. <p>Los cuatro valores aceptan el valor de <code><medida></code> (también en negativo) o <code>'auto'</code>. El valor <code>'auto'</code> significa que un borde dado de la zona de recorte será igual que el límite del borde de la caja generada del elemento (es decir, <code>'auto'</code> significa <code>'0'</code> para <code><top></code> y <code><left></code> en texto <code>'ltr'</code>, igual que el valor computado de la altura más la suma de los anchos verticales del relleno y del borde para <code><bottom></code>, e igual que el valor computado de el ancho más la suma de los anchos horizontales del relleno y del borde para <code><right></code>, así que cuatro valores <code>'auto'</code> causan que la zona de recorte sea igual al borde de la caja del elemento). Cuando las coordenadas se redondean a los coordenadas en píxel, se debe tener cuidado de que ningún píxel permanezca visible cuando <code><left></code> y <code><right></code> tengan el mismo valor (o <code><top></code> y <code><bottom></code> tengan el mismo valor), y recíprocamente que ningún píxel permanezca oculto dentro del borde de la caja del elemento cuando estos valores son <code>'auto'</code>.</p>

La zona de recorte de un elemento recorta cualquier aspecto del elemento (por ejemplo, contenido, hijos, fondos, bordes, decoración del texto, contorno, etc.) que está fuera de la zona de recorte. Los ascendentes del elemento pueden también recortar porciones de su contenido (por ejemplo, vía sus propiedades `'clip'` y/o si la propiedad `'overflow'` no es `'visible'`); lo que se procesa es la intersección de las distintas zonas de recorte.

Si la zona de recorte excede los límites de la ventana del documento de la AU, el contenido se puede recortar a esa ventana por el entorno operativo nativo.

VISIBILIDAD ('VISIBILITY')

La propiedad `'visibility'` especifica si las cajas generadas por un elemento son procesadas, y aunque sean invisibles siguen afectando la composición. Hay que utilizar la propiedad `'display'` como `'none'` para suprimir completamente la generación de una caja.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'visibility'	visible hidden collapse inherit	visible	Todos los elementos	Si	N/A	Visual	2
Valor computado:	Según el especificado						

Valores posibles para **'visibility'**:

Nombre	Explicación
visible	La caja generada es visible.

Nombre	Explicación
hidden	La caja generada es invisible (completamente transparente), pero sigue afectando la composición. Además, los descendientes del elemento serán visibles si tienen <code>'visibility: visible'</code> .
collapse	Si se usa en otros elementos que no sean filas, grupos de filas, columnas o grupos de columnas, <code>'collapse'</code> tiene el mismo significado que <code>'hidden'</code> .

10. Contenido generado, numeración automática y listas



PSEUDO-ELEMENTOS :BEFORE Y :AFTER

Los autores especifican el estilo y la ubicación del contenido generado con los pseudo-elementos **:before** y **:after**.

Los pseudo-elementos **:before** (antes) y **:after** (después) permiten especificar la ubicación y el estilo de un contenido generado antes y después del contenido de la estructura del documento de un elemento. La propiedad `'content'`, junto con estos pseudo-elementos, especifica lo que se inserta.

Por ejemplo, el siguiente código inserta la cadena "Importante:" antes del contenido de cada elemento **P** con un `class="important"`:

```
p.important:before { content: "Importante: "; }
p.important { font-weight: bold; color: red; }
```

El formato aplicado a un elemento, incluye también al contenido generado para el mismo.

Por lo tanto, la fuente en negrita y rojo se aplicará también a la palabra "Importante:"

Los pseudo-elementos **:before** y **:after** heredan cualquier propiedad que se puede heredar del elemento en la estructura del documento al cual están ligados.

En el siguiente ejemplo, los elementos **Q** serán en color gris, y se le insertarán comillas de apertura antes de cada uno, las cuáles serán en color azul:

```
q { color: gray; }
q:before { content: open-quote; color: blue; }
```

En la declaración del pseudo-elemento **:before** o **:after**, las propiedades no-heredadas toman sus valores iniciales.

En base a esto debemos considerar para el ejemplo anterior, que 'padding' ha tomado su valor inicial `'0'` y en

caso de querer cambiarlo habrá que especificarlo con otro valor.

LA PROPIEDAD 'CONTENT'

La propiedad `'content'` se utiliza con los pseudo-elementos `:before` y `:after` para generar el contenido en un documento.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'content'	normal [<cadena> <uri> <contador> attr(<identificador>)] o penquote close- quote no-open- quote no-close- quote + inherit	normal	Pseudo- elementos :before y :after	No	N/A	Todos	2
Valor computado:	Para los valores de URI, el URI absoluto; para los valores de attr(), la cadena resultante; si no según el especificado						

Valores posibles para **'content'**:

Nombre	Explicación
normal	El pseudo-elemento no se genera.
<cadena>	Contenido del texto.
<uri>	El valor es un URI que señala un recurso externo. Si una aplicación de usuario no puede soportar el recurso debido a los tipos de medios que soporta, debe ignorar el recurso.
<contador>	Los contadores se pueden especificar con dos funciones distintas: <code>'counter()'</code> o <code>'counters()'</code> . <code>counter()</code> toma dos formas: <code>'counter(nombre)'</code> o <code>'counter(nombre, estilo)'</code> . El texto generado es el valor del contador nombrado a este punto en la estructura del formato; se ajusta al formato del estilo indicado (<code>'decimal'</code> por defecto). <code>counters()</code> también tiene dos formas: <code>'counters(nombre, cadena)'</code> o <code>'counters(nombre, cadena, estilo)'</code> . El texto generado es el valor de todos los contadores con el nombre dado a este punto en la estructura del formato, separado por la cadena especificada. Los contadores se procesan con el estilo indicado (<code>'decimal'</code> por defecto).
'open-quote' 'close-quote'	Estos valores son reemplazados por la cadena apropiada de la propiedad <code>'quotes'</code> .
'no-open-quote' 'no-close-quote'	Igual que <code>'none'</code> , pero incrementa (disminuye) el nivel de la anidamiento de las comillas.
attr(X)	Esta función devuelve como cadena el valor del atributo <code>X</code> para el sujeto del selector. La cadena no es analizada por el procesador de CSS. Si el sujeto del selector no tiene un atributo <code>X</code> , se vuelve una cadena vacía. La distinción entre mayúsculas y minúsculas en el nombre de los atributos depende del lenguaje del documento.

La propiedad `'display'` controla si el contenido está puesto en una caja de bloque, en línea, o marcador.

La siguiente regla coloca la cadena *"Tema:"* antes de cada **H1**:

```
h1:before {
  contenido: "Tema: ";
  display: inline; }
```

Se pueden incluir nuevas líneas en el contenido generado escribiendo la secuencia de escape “\A” en una de las cadenas después de la propiedad `'content'`. Este salto de línea insertado está sujeto a la propiedad `'white-space'` (espacio en blanco).

```
h1:before {
display: block;
white-space: pre;
content: "Tema\A Subject"
}
```

El contenido generado no altera la estructura del documento. En particular, no retroalimenta el procesador del lenguaje del documento (ej., para otro análisis).

COMILLAS

Es posible especificar, de un modo sensible al estilo y dependiente del contexto, como las aplicaciones de usuario deben procesar las comillas. La propiedad `'quotes'` especifica pares de comillas para cada nivel de comillas incrustadas. La propiedad `'content'` da acceso a esas comillas y provoca que sean insertadas antes y después de una cita.

Especificación con la propiedad `'quotes'`

Esta propiedad especifica las comillas para cualquier cantidad de citas incrustadas.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
<code>'quotes'</code>	[<cadena> <cadena>]+ none inherit	depende de la AU	Todos los elementos	Si	N/A	Visual	2
Valor computado:	Según el especificado						

Valores posibles para `'quotes'`:

Nombre	Explicación
<code>none</code>	Los valores <code>'open-quote'</code> y <code>'close-quote'</code> de la propiedad <code>'content'</code> no provocan ninguna comilla.
[<cadena> <cadena>]+	Los valores para los valores de <code>'open-quote'</code> y <code>'close-quote'</code> de la propiedad <code>'content'</code> son tomados de esta lista de pares de comillas (de apertura y de cierre). El primer par (más a la izquierda) representa el nivel exterior de la cita, el segundo par el primer nivel de incrustación, etc. La aplicación de usuario debe aplicar el par de comillas apropiado según el nivel de incrustación.

Veamos que el siguiente código CSS especifica dos pares de comillas para los elementos `q`, que deben colocarse antes y después del contenido del elemento.

```
q { quotes: '""' '""' '""' '""' }
q:before { content: open-quote; }
q:after { content: close-quote; }
```

Veamos el código XHTML al cuál puede aplicarse:

```
<p>Estefanía dijo, <q>Mi padre decía, <q>Los libros son las abejas que llevan el
polen de una inteligencia a otra.</q> Creo que escribía libros para sentirse parte
de la polinización de las mentes.</q></p>
```

Esto daría como resultado:

Estefanía dijo, “Mi padre decía, ‘Los libros son las abejas que llevan el polen de una inteligencia a otra.’ Creo que escribía libros para sentirse parte de la polinización de las mentes.”

Nota: Las comillas anteriores se encuentran en los teclados, sin embargo, se pueden necesitar otros tipos de comillas de los caracteres de la ISO 10646, tales como:

Carácter	Procesada como:	Código de la ISO 10646 (hex)	Descripción
"	"	0022	COMILLAS [comillas dobles en ASCII]
'	'	0027	APÓSTROFE [comilla simple en ASCII HACIA LA IZQUIERDA
<	<	2039	COMILLA ANGULAR SIMPLE HACIA LA IZQUIERDA
>	>	203A	COMILLA ANGULAR SIMPLE HACIA LA DERECHA
«	«	00AB	COMILLAS ANGULARES DOBLES HACIA LA IZQUIERDA
»	»	00BB	COMILLAS ANGULARES DOBLES HACIA LA DERECHA
‘	`	2018	COMILLA DE CIERRE SIMPLE IZQUIERDA [simple alta-9]
’	'	2019	COMILLA DE CIERRE SIMPLE DERECHA [simple alta-9]
“	``	201C	COMILLA DE CIERRE DOBLE IZQUIERDA [simple alta-9]
”	"	201D	COMILLA DE CIERRE DOBLE DERECHA [simple alta-9]
„	“	201E	COMILLAS DOBLES BAJA-9 [doble [doble alta-9

Inserción con la propiedad 'content'

Las comillas son insertadas en los lugares correspondientes de un documento con los valores `'open-quote'` y `'close-quote'` de la propiedad `'content'`. Cada aparición de `'open-quote'` o `'close-quote'` es substituido por una de las cadenas del valor de `'quotes'`, en base a la profundidad del anidado.

'Open-quote' se refiere a la primera de un par de comillas y 'close-quote' a la segunda.

Qué par de comillas se utiliza depende del nivel de anidamiento de las comillas: el número de apariciones de '**open-quote**' en todo el texto generado antes de la presente, menos el número de apariciones de '**close-quote**'. Si la profundidad es 0, se utiliza el primer par, si la profundidad es 1, se utiliza el segundo par, etc. Si la profundidad es mayor que el número de pares, se repite el último par.

Las palabras clave `'no-open-quote'` y `'no-close-quote'` disminuyen el nivel de las citas pero no insertan una comilla.

CONTADORES Y NUMERACIÓN AUTOMÁTICA

La numeración automática es controlada con las propiedades `'counter-increment'` y `'counter-reset'` que son utilizadas con las funciones `counter()` y `counters()` de la propiedad `'content'`.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'counter-reset'	[<identificador> <entero>?]+ none	none	Todos los elementos	No	N/A	Todos	2
'counter-increment'	inherit						

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
Valor computado:	Como el especificado						

La propiedad '`counter-increment`' acepta uno o más nombres de contadores (identificadores), cada uno seguido opcionalmente por un entero. El entero indica en cuanto se incrementa el contador con cada aparición del elemento. El incremento por defecto es 1. Los enteros cero y negativo son admitidos.

La propiedad '`counter-reset`' también contiene una lista de uno o más nombres de contadores, cada uno seguido opcionalmente por un entero. El entero da el valor que en el contador es colocado con cada aparición del elemento. El valor por defecto es 0.

Si '`counter-increment`' se refiere a un contador que no está en el **área de alcance** (ver abajo) de algún '`counter-reset`', se asume que el contador ha sido reseteado a cero por el elemento raíz.

Este ejemplo muestra una forma de numerar capítulos y secciones con "Capítulo 1", "1.1", "1.2", etc.

```
h1:before {
  content: "Capítulo " counter(capítulo) ". ";
  counter-increment: capítulo; /* Añade 1 a capítulo */
  counter-reset: Sección; /* Pone la sección a 0 */
}
h2:before {
  content: counter(capítulo) "." counter(Sección) " ";
  counter-increment: Sección;
}
```

Si un elemento incrementa/resetea un contador y también lo usa (en la propiedad '`content`' de su pseudo- elemento `:before` o `:after`), el contador se utiliza después de ser incrementado/reseteado.

Si un elemento resetea e incrementa un contador, el contador es reseteado primero y luego incrementado.

La propiedad '`counter-reset`' sigue las reglas de cascada.

Contadores anidados y área de alcance

Los contadores son "autoanidados", en el sentido de que la re-utilización de un contador en un elemento hijo automáticamente crea una nueva instancia del contador. Esto es importante en situaciones como las listas de HTML, donde los elementos pueden ser anidados dentro de sí mismo hasta una profundidad arbitraria. Resultaría imposible definir los contadores con un nombre único para cada nivel.

Así, lo siguiente es suficiente para numerar los elementos (ítems) de una lista anidada. El resultado es muy similar al de poner '`display: list-item`' y '`list-style: inside`' en el elemento **LI**:

```
ol { counter-reset: item }
li { display: block }
li:before { content: counter(item) ". "; counter-increment: item }
```

El autoanidamiento se basa en el principio de que cada elemento que tiene '`counter-reset`' para un contador *X*, crea un nuevo contador *X*, el área de alcance del cual es el elemento, sus hermanos siguientes y todos los descendientes del elemento y sus siguientes hermanos. En el ejemplo de arriba, un **OL** creará un contador, y todos los hijos de **OL** se referirán a ese contador.

La siguiente hoja de estilos numera los elementos (ítems) de una lista como "1", "1.1", "1.1.1", etc.

```
ol { counter-reset: item }
li { display: block }
li:before { content: counters(item, ".") " "; counter-increment: item }
```

Estilos de contadores

Por defecto, los contadores son procesados con números decimales, pero todos los estilos disponibles para la propiedad `'list-style-type'` están también disponibles para los contadores. La sintaxis es: `counter(nombre)` para el estilo por defecto, o `counter(nombre, 'list-style-type')`.

Todos los estilos están permitidos, incluyendo `'disc'`, `'circle'`, `'square'`, y `'none'`.

```
h1:before { content: counter(chno, upper-latin) ". " }
h2:before { content: counter(Sección, upper-roman) " - " }
blockquote:after { content: " [" counter(bq, hebrew) "]" }
div.note:before { content: counter(notecntr, disc) " " }
p:before { content: counter(p, none) }
```

Contadores en elementos con `'display: none'`

Un elemento que no es mostrado (`'display'` fijado a `'none'`) no puede incrementar o resetear un contador.

Por ejemplo, con la siguiente hoja de estilo, los **h2** de la clase "oculto" no incrementan a 'conteo'.

```
h2.oculto {counter-increment: conteo; display: none}
```

Los elementos con `'visibility'` puesto a `'hidden'`, por otro lado, si incrementan los contadores.

LISTAS

Un elemento con `'display: list-item'` genera una caja principal para el contenido de los elementos y un marcador opcional de caja como una indicación visual de que el elemento es una lista de objetos.

Las propiedades de lista describen un formato visual básico de listas: estas permiten a hojas de estilo especificar el tipo de marcador (imagen, glyph, o número), y la posición del marcador con respecto a la caja principal (fuera de ella o dentro de ella antes del contenido). No permiten a los autores especificar distintos estilos (colores, fuentes, alineación, etc.) para la lista de marcadores o ajustar su posición con respecto a la caja principal, estos pueden ser derivados de la caja principal.

Las propiedades del fondo se aplican solo a la caja principal, un marcador de caja `'outside'` es transparente.

`'list-style-type'`

Esta propiedad especifica la apariencia del marcador de los elementos (items) de la lista si `'list-style-image'` tiene el valor `'none'` o si la imagen señalada por el URI no puede ser mostrada.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'list-style-type'	disc circle square decimal decimal-leading-zero lower-roman upper-roman lower-greek lower-latin upper-latin armenian georgian none inherit	disc	Elementos con <code>'display: list-item'</code>	Si	N/A	Visual	1

El valor `'none'` especifica ningún marcador, de lo contrario hay 3 tipos de marcadores: glyphs, sistemas numéricos, y sistemas alfabéticos.

Valores posibles para `'list-style-type'`:

Tipo	Nombre	Explicación
Glyphs	disc	El procesamiento exacto depende de la aplicación de usuario.
	circle	
	square	
Sistemas numéricos	decimal	Número decimal, comenzando con 1.
	decimal-leading-zero	Números decimales con ceros iniciales (ej., 01, 02, 03, ..., 98, 99).
	lower-roman	Números romanos en minúsculas (i, ii, iii, iv, v, etc.).
	upper-roman	Números romanos en mayúsculas (I, II, III, IV, V, etc.).
	georgian	Numeración georgiana tradicional (an, ban, gan, ..., he, tan, in, ...).
	armenian	Numeración armenia tradicional.
Sistemas alfabéticos	lower-latin, lower-alpha	Letras minúsculas en ascii (a, b, c, ... z).
	upper-latin, upper-alpha	Letras mayúsculas en ascii (A, B, C, ... Z).
	lower-greek	Minúsculas griegas clásicas alpha, beta, gamma, ... (α, β, γ, ...)

Una aplicación de usuario que no reconoce un sistema numérico debe utilizar el `'decimal'`.

Al no haberse definido como se desenvuelven los sistemas alfabéticos al final del alfabeto, se recomienda utilizar números para listas extensas.

Por ejemplo, el siguiente documento HTML:

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

<html>

  <head>
    <title> Numeración con latinas minúsculas </title>
    <style type="text/css">
      ol { list-style-type: lower-roman }
    </style>
  </head>

  <body>
    <ol>
      <li>Primer elemento.</li>
      <li>Segundo elemento.</li>
      <li>Tercer elemento.</li>
    </ol>
  </body>
</html>
```

Puede producir algo así:

i Primer elemento.
ii Segundo elemento.
iii Tercer elemento.

`'list-style-image'`

Esta propiedad define la imagen que será utilizada como marcador de elementos (items) de una lista. Cuando la imagen está disponible, reemplazará el marcador definido con la propiedad `'list-style-type'`.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'counter-reset'	<uri> none inherit	none	Elementos con 'display: list-item'	Si	N/A	Visual	1
Valor computado:	URI absoluta o 'none'						

En el siguiente ejemplo, la imagen vineta.gif será el marcador al comienzo de cada elemento:

```
ul { list-style-image: url(vineta.gif); }
```

'list-style-position'

Esta propiedad especifica la posición de la caja marcador en la caja de bloque principal.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'list-style-position'	inside outside inherit	outside	Elementos con 'display: list-item'	Si	N/A	Visual	1
Valor computado:	Como el especificado.						

Valores posibles para 'list-style-position':

Nombre	Explicación
outside	La caja del marcador está fuera de la caja de bloque principal.
inside	La caja del marcador está en la primera caja en línea en la caja de bloque principal, después de la cual fluye el contenido del elemento.

Por ejemplo:

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

<html>

  <head>
    <title>Numeración con latinas minúsculas</title>
    <style type="text/css">
      ul { list-style-type: outside }
      ul.compact {list-style:inside }
    </style>
  </head>

  <body>
    <ul>
      <li>Primer elemento</li>
      <li>Segundo elemento</li>
    </ul>
    <ul class="compact">
      <li>Primer elemento</li>
      <li>Segundo elemento</li>
    </ul>
  </body>
</html>
```

'list-style'

La propiedad 'list-style' es una notación corta para definir las tres propiedades 'list-style-type', 'list-style-image', y 'list-style-position' en el mismo lugar en la hoja de estilo.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'list-style'	[<'list-style-type'> <'list-style-position'> <'list-style-image'>] inherit	ver propiedades individuales	Elementos con with 'display: list-item'	Si	N/A	Visual	1
Valor computado:	Ver propiedades individuales.						

```
ul { list-style: upper-roman inside }
ul > li > ul { list-style: circle outside }
```

La información sobre '**list-style**' puede aplicarse sobre cada elemento de lista (ítems), sin embargo esto puede generar problemas. Las siguientes reglas parecen similares, pero la primera declara un selector descendiente y la segunda un (más específico) selector hijo.

```
ol.prueba li { list-style: disc } /* cualquier "li" descendiente de "ol" */
ol.prueba > li { list-style: disc } /* cualquier "li" hijo de un "ol" */
```

Con el selector de descendientes pueden no lograrse los resultados esperados, ya que el orden de cascada asigna a los descendientes las clases de los antepasados. Con las siguientes reglas se soluciona el problema empleando en cambio un selector hijo:

```
ol.prueba > li { list-style: lower-alpha }
ul li { list-style: disc }
```

Sin embargo, otra solución sería especificar la información sobre '**list-style**' sólo en los elementos de tipo lista:

```
ol.prueba { list-style: lower-alpha }
ul { list-style: disc }
```

La herencia transferirá los valores de '**list-style**' de los elementos **OL** y **UL** a los elementos **LI**.

Un valor de '**none**' para la propiedad '**list-style**' pone a '**list-style-type**' y '**list-style-image**' como '**none**':

```
ul { list-style: none }
```

11. Medios paginados



Los medios paginados (ej., papel, transparencias, páginas que se muestran en la pantalla, etc.) se diferencian de los medios continuos en que el contenido del documento está dividido en una o más páginas discretas.

La aplicación de usuario se encarga de transferir las cajas de página de un documento sobre las hojas reales donde el documento será procesado en última instancia (papel, transparencia, pantalla, etc.).

CAJAS DE PÁGINA (REGLA @PAGE)

La **caja de página** es una región rectangular que contiene dos áreas:

- El **área de la página**: El área de la página incluye las cajas presentadas en esa página. Los límites del área de la página actúan como el bloque de contenido inicial de la disposición efectuada entre saltos de página.
- El **área del margen**, que rodea al área de la página y que puede especificarse dentro de una regla @page.

Una regla **@page** consiste en la palabra clave "@page", seguida por un selector opcional de la página y por un bloque de declaraciones. Las declaraciones en una regla @page serían en el **contexto de la página**. El **selector de la página** especifica a qué páginas se aplican las declaraciones (la primera página, todas las páginas izquierdas, o todas las páginas derechas).

Márgenes de la página

En CSS 2.1, solamente las propiedades del margen ('margin-top', 'margin-right', 'margin-bottom', 'margin-left', y 'margin') se aplican dentro del contexto de la página.

En la siguiente imagen podemos observar las relaciones entre la hoja, la caja de página y el margen de la página. Hay que tener en cuenta que una caja de página puede ser transferida a una variedad de hojas con diferentes dimensiones y proporciones (A4, A5, letter, office, etc.)



El siguiente ejemplo fija los cuatro márgenes de una página en 3cm:

```
@page { margin: 3cm; }
```

El contexto de la página no tiene ninguna noción de fuentes, por lo que las unidades '*em*' y '*ex*' no están permitidas. Los valores de porcentajes en las propiedades del margen son relativos a las dimensiones de la caja de página; para los márgenes izquierdo y derecho, refieren al ancho de la caja de página mientras que para los márgenes superior e inferior, refieren a la altura de la caja de página.

Debido a los valores negativos del margen (en la caja de página o en elementos) o del posicionamiento absoluto, el contenido puede terminar fuera de la caja de página, sin embargo este contenido puede ser "cortado" -- por la aplicación de usuario, la impresora, o en última instancia, la guillotina de papel.

El valor computado de los márgenes de la caja para la parte superior o inferior es cero.

Si una caja de página no se ajusta a las dimensiones de la hoja de destino, la aplicación del usuario (consultando al usuario) puede:

- Rotar 90° la caja de página si esto hace que la página se ajuste.
- Escalar la página para ajustarla al destino.

Cuando la caja de página es más pequeña que el tamaño del destino, la aplicación de usuario es libre de ubicar la caja de página en cualquier lugar de la hoja. Sin embargo, se recomienda que la caja de página esté centrada en la hoja puesto que así se alinearán las páginas de doble cara y evitará la pérdida accidental de información que es impresa cerca del borde de la hoja.

Selectores de la página ('*:left*', '*:right*' y '*:first*')

Al imprimir documentos de doble cara, la caja de página en páginas izquierdas y derechas puede ser diferente. Con las pseudo-clases '*:left*' y '*:right*' las páginas son clasificadas automáticamente por la aplicación de usuario en páginas izquierdas o derechas. Además, con la pseudo-clase '*:first*' se pueden fijar márgenes diferentes para la primer página de un documento.

Veamos la aplicación de estas pseudo-clases:

```
@page { margin: 2cm; }  
@page :left { margin-left: 3cm; margin-right: 2cm; }  
@page :right { margin-left: 2cm; margin-right: 3cm; }  
@page :first { margin-top: 8cm; }
```

Hay que recordar que las propiedades especificadas en una regla `@page :first` reemplazan las especificadas en las

reglas @page :left o :right.

Las declaraciones de márgenes sobre paginas izquierda (left), derecha (right) y primera (first) pueden causar anchos diferentes en el área de la página. En ese caso, las aplicaciones de usuario pueden utilizar un sólo ancho en el área de la página en páginas izquierdas, derechas, y primeras, el cuál estará basado en el ancho de la primer página.

Contenido fuera de la caja de la página

Al estructurar el contenido en el modelo de la página, parte del contenido puede terminar fuera de la caja de página (generalmente con cajas posicionadas de manera absoluta).

Si bien el procedimiento frente a contenido fuera de la caja de página no está establecido se recomienda tener en cuenta lo siguiente:

- El contenido debe ser admitido ligeramente fuera de la caja de la página para permitir el "sangrado" de las páginas.
- Las aplicaciones de usuario deben evitar generar una gran cantidad de cajas de página vacías para respetar la ubicación de los elementos (ej., nadie quiere imprimir 100 páginas en blanco).
- Los autores no deben posicionar elementos en localizaciones inconvenientes sólo para evitar procesarlos.
- Las aplicaciones de usuario pueden manipular las cajas ubicadas fuera de la caja de página de varias maneras, incluyendo el descarte de las mismas o crear las cajas de página para ellas al final del documento.

SALTOS DE PÁGINA

Hay cinco propiedades que le indican a la aplicación de usuario donde producir los saltos de página, y en qué página (izquierda o derecha) debe continuar el contenido siguiente. Cada salto de página termina la composición de la caja de página actual y provoca que las partes restantes de la estructura del documento sean colocadas en una nueva caja de página.

Salto de página ('page-break-before', 'page-break-after' y 'page-break-inside')

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
‘page-break-before’	auto always avoid left right inherit	auto	Elementos a nivel de bloque (block-level elements)	No	N/A	Visual, paginados	2
‘page-break-after’							
‘page-break-inside’	void auto inherit			Si			
Valor computado:	Como el especificado.						

Valores posibles para 'page-break-before', 'page-break-after' y 'page-break-inside':

Nombre	Explicación
auto	No fuerza ni prohíbe un salto de página antes de (después de, en) la caja generada.
always	Siempre fuerza un salto de página antes (después) de la caja generada.
avoid	Evita un salto de página antes (después de, en) la caja generada.
left	Fuerza uno o dos saltos de páginas antes (después) de la caja generada para ajustar el formato la página siguiente como una página izquierda.
right	Fuerza uno o dos saltos de página antes (después) de la caja generada para ajustar el formato de la página siguiente como una página derecha.

Si la primera página de un documento es `:left` o `:right` depende de la dirección principal de la escritura del documento. Una aplicación de usuario con conformidad puede interpretar los valores `'left'` y `'right'` como `'always'`.

Una localización potencial de un salto de página está típicamente bajo la influencia de la propiedad `'page-break-inside'` del elemento padre, la propiedad `'page-break-after'` del elemento precedente y la propiedad `'page-break-before'` del elemento siguiente. Cuando estas propiedades tienen valores distintos de `'auto'`, los valores `'always'`, `'left'`, y `'right'` tiene prioridad sobre `'avoid'`.

Estas propiedades se aplican solamente a los elementos a nivel de bloque que están en el flujo normal del elemento raíz.

Con el siguiente ejemplo, todos los títulos H1 comenzarán en una nueva página:

```
h1 { page-break-before: always; }
```

Saltos dentro de los elementos ('orphans' y 'widows')

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'orphans'	<entero> inherit	2	Elementos a nivel de bloque (block-level elements)	Si	N/A	Visual, paginados	2
'widows'							
Valor computado:	Como el especificado.						

La propiedad `'orphans'` especifica el número mínimo de líneas de un párrafo que se deben dejar al final de una página. La propiedad `'widows'` especifica el número mínimo de líneas de un párrafo que se deben dejar al comienzo de una página.

Veamos el siguiente código:

```
p { orphans: 4; widows: 2; }
```

De existir 19 líneas disponibles al final de la página actual:

Si un párrafo al final de la página contiene 19 líneas o menos, este debe colocarse completamente en la página actual.

Si un párrafo contiene 20 o 21 líneas, como se debe cumplir con dos líneas por la propiedad `'widows'`, la página siguiente debe contener dos líneas.

Si el párrafo contiene 22 líneas o mas, el final de la primera página debe contener 19 líneas y el comienzo de la segunda página el resto de las líneas.

CASCADA EN EL CONTEXTO DE PÁGINA

Las declaraciones en el contexto de página obedecen la cascada.

En el siguiente ejemplo, el margen de todas las páginas será de 2cm, sin embargo, por la mayor especificidad de la pseudo-clase, el margen derecho de las páginas derechas será sobrescrito a 3cm.

```
@page { margin-left: 2cm; }
@page :right { margin-right: 3cm; }
```

12. Colores y fondos



CSS permite especificar el color del primer plano y el del fondo (que puede ser un color o una imagen) de un elemento. Al colocar una imagen de fondo se puede especificar si la misma debe repetirse, y si debe quedar fija con respecto al acceso visual o desplazarse junto con el documento.

Recomendamos ver la sección sobre [colores](#) para identificar los valores válidos.

COLOR DEL PRIMER PLANO ('COLOR')

La propiedad `'color'` define el color del primer plano del contenido del texto de un elemento.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
<code>'color'</code>	<color> inherit	depende de la UA	Todos los elementos.	Si	N/A	Visual	1
Valor computado:	Como el especificado.						

Formas de especificarlo:

```
p { color: rojo; }  
p { color: rgb (255,0,0); }
```

🔗 [Verificar la aplicación de color al texto de diferentes elementos \(CD > ejemplos > css > color_fondo > color.html\)](#)

FONDO

Se puede especificar el fondo de un elemento como un color o una imagen. Según el modelo de caja, el `'background'`

afecta el fondo del contenido, relleno y borde (colores y estilos fijados con las propiedades del borde) y no a los márgenes que son siempre transparentes.

Las propiedades del fondo no se heredan, pero el fondo de la caja del padre se verá debido al valor inicial `'transparent'` para `'background-color'`. El fondo del elemento raíz (**HTML**) se convierte en el fondo del lienzo y lo cubre completamente.

Para los documentos HTML, se recomienda especificar el fondo para **BODY** y no para **HTML**. Para las aplicaciones de usuario si el valor de la propiedad `'background'` para HTML es distinto a `'transparent'` entonces lo utiliza, de otro modo se usa el valor de la propiedad `'background'` para el elemento **BODY**. Si el valor resultante es `'transparent'`, el procesamiento es indefinido. Esto no se aplica a los documentos XHTML.

Veamos la aplicación de un fondo con la imagen fondo.jpg y el color azul (que se mostrará si la imagen no se puede procesar):-

```
body { background: blue url("fondo.jpg") }
```

El fondo de los elementos que forman un contexto de apilamiento (véase la propiedad `'z-index'`) se pintan en el fondo del contexto de apilamiento del elemento, debajo de cualquier contenido del contexto de apilamiento.

Propiedades del fondo

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'background-color'	<color> transparent inherit	transparent	Todos los elementos.	No	N/A	Visual	1
'background-image'	<uri> none inherit	none					
'background-repeat'	repeat repeat-x repeat-y no-repeat inherit	repeat					
'background-attachment'	scroll fixed inherit	scroll					
'background-position'	[[<porcentaje> <medida> left center right] [<porcentaje> <medida> top center bottom]?] [left center right] [top center bottom]] inherit	0% 0%			Referido al tamaño de la propia caja		
'background'	[<'background-color'> <'background-image'> <'background-repeat'> <'background-attachment'> <'background-position'>] inherit	ver propiedades individuales			Permitido en la propiedad <code>'background-position'</code>		

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
Valor computado:	para <code>'background-color'</code> , <code>'background-repeat'</code> y <code>'background-attachment'</code> : Como el especificado para <code>'background-image'</code> : URI absoluta para <code>'background-position'</code> : para <code><medida></code> el valor absoluto, sino un porcentaje para <code>'background'</code> : vea las propiedades individuales						

'background-color'

Fija el color de fondo de un elemento, ya sea con un valor `<color>` o con la palabra clave `'transparent'` (hace que los colores subyacentes de vean a través del objeto):

```
h1 { background-color: black; }
```

'background-image'

Esta propiedad fija la imagen de fondo de un elemento, en cuyo caso es recomendado fijar también el color de fondo que será utilizado si la imagen no está disponible. Cuando la imagen está disponible, se procesa encima del color de fondo (el cuál es visible a través de las partes transparentes de la imagen).

Se puede especificar cualquier `< uri >` para indicar la localización de la imagen, o `'none'` para no utilizar ninguna imagen.

```
body { background-image: url("fondo.jpg"); }
p { background-image: none; }
```

'background-repeat'

Esta propiedad especifica si una imagen de fondo es repetida (mosaico) o no y de que modo. El mosaico cubre las áreas de contenido, relleno y borde de una caja.

Valores posibles para 'background-repeat':

Nombre	Explicación
repeat	La imagen se repite horizontal y verticalmente.
repeat-x	La imagen se repite solo horizontalmente.
repeat-y	La imagen se repite solo verticalmente.
no-repeat	La imagen no se repite y por lo tanto, se dibuja una sola copia de la imagen.

Veamos algunos códigos de aplicación:

```
body {
  background: white url("fondo.jpg");
  background-repeat: repeat-y; }
```

'background-attachment'

Cuando hay una imagen de fondo esta propiedad especifica si la misma es fija con respecto al acceso visual (`'fixed'`) o se desplaza junto con el bloque que contiene (`'scroll'`). Si un elemento tiene un mecanismo de desplazamiento el fondo `'fixed'` no se mueve con el elemento, y un fondo `'scroll'` no se mueve con el mecanismo de desplazamiento.

Este ejemplo crea una banda vertical que permanece fija en el acceso visual cuando el elemento es desplazado.

```
body {
background: green url("fondo.jpg");
background-repeat: repeat-y;
background-attachment: fixed;
}
```

Las aplicaciones de usuario que no soportan fondos **'fixed'** (debido a limitaciones) deben ignorar las declaraciones con la palabra clave **'fixed'**.

'background-position'

Si se ha especificado una imagen de fondo, esta propiedad especifica su posición inicial.

Valores posibles para **'background-position'**:

Nombre	Explicación
<porcentaje> <porcentaje>	Con un par de valores de '0% 0%', la esquina superior izquierda de la imagen se alinea con la esquina superior izquierda del límite del relleno de la caja. Un par de valores de '100% 100%' pone la esquina inferior derecha de la imagen en la esquina inferior derecha del área de relleno. Con un par de valores de '14% 84%', el punto en la imagen debe ser colocado en el punto 14% del costado y a un 84% hacia abajo en el área de relleno.
<medida> <medida>	Con un par de valores de '2cm 2cm', la esquina superior izquierda de la imagen se pone 2cm a la derecha y 2cm debajo de la esquina superior izquierda del área de relleno.
top left y left top	Igual que '0% 0%'.
top, top center y center top	Igual que '50% 0%'.
right top y top right	Igual que '100% 0%'.
left, left center y center left	Igual que '0% 50%'.
center y center center	Igual que '50% 50%'.
right, right center y center right	Igual que '100% 50%'.
bottom left y left bottom	Igual que '0% 100%'.
bottom, bottom center y center bottom	Igual que '50% 100%'.
bottom right y right bottom	Igual que '100% 100%'.

Si se da solamente un valor de porcentaje o de medida, éste fija solo la posición horizontal ya que la posición vertical será el 50%. Si se dan dos valores, la posición horizontal viene primero. Se permiten combinaciones de palabra clave, medida y porcentaje se permiten, (Ej., '50% 2cm' o 'center 2cm' o 'center 10%'), pero en ese caso **'left'** y **'right'** solo pueden ser utilizados como el primer valor, y **'top'** y **'bottom'** como segundo valor. Se permiten las posiciones negativas.

```
body { background: url("fondo.jpg") right top } /* 100% 0% */
body { background: url("fondo.jpg") center 0% } /* 50% 0% */
body { background: url("fondo.jpg") center } /* 50% 50% */
body { background: url("fondo.jpg") bottom } /* 50% 100% */
```

'background'

La propiedad **'background'** es una propiedad resumida para fijar las propiedades individuales del fondo (es decir, **'background-color'**, **'background-image'**, **'background-repeat'**, **'background-attachment'** y **'background-position'**) en una sola propiedad. La propiedad **'background'** primero fija todas las propiedades con sus valores iniciales, y luego asigna los valores explícitamente asignados en la declaración.

En **BODY** se ha especificado sólo el **'background-color'** y las otras propiedades individuales se fijan con su

valor inicial. En **P** se han especificado todas las propiedades individuales:

```
body { background: #FF0000; }  
p { background: #CCC url("rayas.gif") 50% repeat fixed; }
```

🔗 [Verificar la aplicación de fondos a diferentes elementos \(CD > ejemplos > css > color_fondo > fondo.html\)](#)

13. Fuentes



Una tarea común es fijar las fuentes a aplicar a un documento y a los elementos dentro del mismo. Desafortunadamente, los términos que se aplican a una familia de fuente pueden no ser apropiados para otras (Ej. 'itálic' se utiliza comúnmente para etiquetar el texto inclinado, así como también *Oblique*, *Slanted*, *Incline*, *Cursive* o *Kursiv*).

FAMILIA DE FUENTES ('FONT-FAMILY')

El valor de esta propiedad es una lista (separada por comas) de prioridades con nombres de familias de fuentes y/o nombres de familia genéricos.

```
body { font-family: Gill, Helvetica, sans-serif }
```

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'font-family'	[[<nombre de la familia> <familia genérica>], <nombre de la familia> <familia genérica>]*] inherit	depende de la UA	Todos los elementos.	Si	N/A	Visual	1
Valor computado:	Como el especificado						

Valores posibles para 'font-family':

Nombre	Explicación
<nombre de la familia>	El nombre de una familia de fuentes de preferencia. Ej. "Gill" y "helvetica" son familias de fuentes.
<familia genérica>	<p>En el ejemplo de arriba, el último valor es un nombre de familia genérico. Los siguientes nombres de familias genéricas se definen:</p> <ul style="list-style-type: none"> • 'serif' (ej. Times) • 'sans-serif' (ej. helvetica) • 'cursivo' (ej. Zapf-Chancery) • 'fantasía' (ej. Western) • 'monospace' (Ej. Courier) <p>Se recomienda ofrecer una familia genérica como última alternativa. Son palabras claves y no pueden ir entre comillas.</p>

Si un nombre de familia que no está entre comillas contiene paréntesis, corchetes y/o llaves, estos deben ser escapados. Los nombres de fuentes que contienen cualquier otro carácter o espacio en blanco deben ir entre comillas.

```
body { font-family: "Times New Roman", serif }
```

Si se omite las comillas, cualquier carácter de espacio en blanco antes y después del nombre de la fuente es ignorado y cualquier secuencia de los caracteres de espacio en blanco dentro del nombre de la fuente se convierte a un solo espacio. Los nombres de familias de fuentes que pueden ser iguales que un valor de una palabra clave (ej. 'initial', 'inherit', 'default', 'serif', 'sansserif', 'monospace', 'fantasy', y 'cursive') se deben poner entre comillas para prevenir la confusión con las palabras claves.

ESTILOS DE FUENTE ('FONT-STYLE')

La propiedad '`font-style`' selecciona entre el aspecto normal, italic y oblique dentro de una familia de fuentes.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'font-style'	normal italic oblique inherit	normal	Todos los elementos.	Si	N/A	Visual	1
Valor computado:	Como el especificado						

Valores posibles para '`font-style`':

Nombre	Explicación
normal	Es el estilo normal de fuente, denominado también como "roman" o "upright". Selecciona una fuente que se clasifica como ' <code>normal</code> ' en la base de datos de fuentes de la AU.
italic	Selecciona una fuente en la base de la AU clasificada como ' <code>italic</code> ', y si no está disponible, una definida como ' <code>oblique</code> '. También son llamadas las fuentes clasificadas como ' <code>cursive</code> ' o ' <code>kursiv</code> '.
oblique	Selecciona una fuente en la base de la AU clasificada como ' <code>oblique</code> ', ' <code>slanted</code> ' o ' <code>incline</code> '. La fuente que definida como ' <code>oblique</code> ' en la base de datos de fuentes de la AU puede haber sido generada en realidad inclinando electrónicamente una fuente normal.

En el siguiente ejemplo, los encabezados serán mostrados con una fuente itálica, y cualquier **EM** dentro de un **H1** con un aspecto normal.

```
h1, h2, h3 { font-style: italic }
h1 em { font-style: normal }
```

SMALL-CAPS ('FONT-VARIANT')

Una variación en una familia de fuentes es **small-caps** (versales o versalitas), donde las letras minúsculas son similares a las mayúsculas, pero en un tamaño más pequeño y con proporciones ligeramente diferentes.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'font-variant'	normal small-caps inherit	normal	Todos los elementos.	Si	N/A	Visual	1
Valor computado:	Como el especificado						

Valores posibles para 'font-variant':

Nombre	Explicación
normal	Selecciona una fuente que no está definida como una fuente small-caps (versalita)
small-caps	Selecciona una fuente de versalita (small-caps). Como un último recurso, la fuente small-caps (versalita) puede ser creada tomando una fuente normal y reemplazando las letras minúsculas por los caracteres mayúsculas reducidos.

En el siguiente ejemplo, los elementos **H2** son mostrados en versalitas, y cualquier elemento **EM** dentro de un **H2** estará en versalitas itálicas.

```
h2 { font-variant: small-caps; }
em { font-style: italic; }
```

GROSOR DE LA FUENTE ('FONT-WEIGHT')

Esta propiedad selecciona el peso de la fuente.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'font-weight'	normal bold bolder lighter 100 200 300 400 500 600 700 800 900 inherit	normal	Todos los elementos.	Si	N/A	Visual	1
Valor computado:	<ul style="list-style-type: none"> Uno de los valores de número legales, o Una de las combinaciones de valores de números legales con uno o más de los valores relativos (bolder o lighter). Este tipo de valor computado es necesario cuando la fuente en cuestión no tiene todas las variantes de peso necesarias. 						

Los valores de '100' a '900' forman una secuencia, donde cada valor indica un peso al menos tan oscuro (grueso) como su predecesor, donde 100 es el más fino y 900 el más grueso.

Luego existen las palabras claves 'lighter', 'normal' (equivale al valor 400), 'bold' (equivale al valor 700) y 'bolder'.

En base a esto podemos decir que los valores declarados a los dos párrafos son equivalentes:

```
p { font-weight: normal; }
p { font-weight: 400; }
```

Los valores 'bolder' y 'lighter' seleccionan los pesos de fuentes con relación al peso heredado del padre. De este modo, los elementos hijos heredan el peso resultante, no el valor de la palabra clave.

```
p { font-weight: 400 }
em { font-weight: bolder }
```

A las fuentes se le asignan nombres descriptivos para diferenciar el peso, que nos están universalmente aceptados. El papel de los mismos es distinguir el grosor dentro de una misma familia de fuentes, y algunos de ellos son: *Regular*, *Roman*, *Book*, *Medium*, *Semi-* o *DemiBold*, *Bold*, or *Black*. Debido a esta falta de estandarización, en CSS se utiliza la escala numérica antes descrita, en la cuál el valor `'400'` corresponde con el aspecto "normal" del texto para esa familia, que también puede tomar los nombres *Book*, *Regular*, *Roman*, *Normal* o a veces *Medium*.

Estos nombres se asignan con la escala numérica en relación con la cantidad de nombres existentes para la familia. Si la familia ya utiliza una escala numérica de nueve valores (como por ej. OpenType), los pesos de la fuente deberán ser mapeados directamente. Algunas veces *Book*, *Regular*, *Roman*, *Normal* o *Medium* equivalen con el valor `'400'`, pero si en una misma familia se presentan las cuatro primeras con *Medium*, entonces a esta última se le asigna el valor `'500'`. La fuente designada con "Bold" corresponde al valor del peso `'700'`.

Si hay menos de 9 pesos en la familia, se deben llenar los "agujeros" hasta completar los 9. Si `'500'` no está asignado, se asignara la misma fuente como `'400'`. Si algunos de los valores `'600'`, `'700'`, `'800'` o `'900'` están sin asignar, estos serán asignados al mismo aspecto que la siguiente palabra clave asignada, si la hay, o el siguiente mas claro si no. Si cualquiera de `'300'`, `'200'` o `'100'` no es asignado, se asignara la palabra clave siguiente mas clara, si la hay, o la siguiente mas oscura si no.

Ya que las palabras claves relativas `'bolder'` y `'lighter'` deben oscurecer o aclarar el aspecto dentro de una familia que puede no tener aspectos alineados con todos los valores de pesos simbólicos, la equivalencia de `'bolder'` es el siguiente aspecto oscuro disponible dentro de la familia y la equivalencia de `'lighter'` es el siguiente aspecto más claro en la familia. Para ser exactos, el significado de las palabras claves relativas `'bolder'` y `'lighter'` son los siguientes:

- **'bolder'** selecciona el siguiente peso asignado a una fuente que es más oscuro que el heredado. Si no existe tal peso, el resultado es el siguiente valor numérico más oscuro (y la fuente permanece sin cambios), a menos que el valor heredado fuese `'900'` en cuyo caso el peso resultante es también `'900'`.
- **'lighter'** selecciona la siguiente palabra clave más clara con una fuente diferente del heredado, a menos que no haya tal fuente, en tal caso selecciona el siguiente valor numérico más claro (y la fuente se mantiene sin cambios).

TAMAÑO DE LA FUENTE ('FONT-SIZE')

El tamaño de la fuente es equivalente con la escala `'em'`, un concepto utilizado en tipografía, tal como lo mostramos en el ejemplo de medidas relativas.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'font-size'	<tamaño-absoluto> <tamaño-relativo> <medida> <porcentaje> inherit	medium	Todos los elementos.	Si	Referido al tamaño de la fuente del elemento padre	Visual	1
Valor computado:	Medida absoluta						

Valores posibles para **'font-size'**:

Nombre	Explicación
<tamaño-absoluto>	Las palabras claves del tamaño absoluto son un índice en una tabla de tamaños de fuentes computadas y guardadas por la aplicación de usuario y los posibles valores son: xx-small (1), x-small, small (2), medium (3), large (4), x-large (5), xx-large (6). Los valores entre paréntesis son tamaños de fuente HTML que proporcionan una referencia a la AU.
<tamaño-relativo>	Una palabra clave de tamaño relativo es interpretada como relativa a la tabla de tamaños de fuente y al tamaño de la fuente del elemento padre. Los posibles valores son: [larger smaller]. Por ejemplo, si el elemento padre tiene un tamaño de fuente 'medium', un valor 'larger' hará que el tamaño de la fuente del elemento actual sea 'large'.

Los valores **<medida>** y **<porcentaje>** no deben tener en cuenta la tabla de tamaños de fuente cuando calculan el tamaño de la fuente de un elemento. Los valores negativos no son permitidos.

En la propiedad `'font-size'` los valores `'em'` y `'ex'` se refieren al tamaño de fuente computado del elemento padre.

Algunos ejemplos podrían ser:

```
p { font-size: 12px; }
strong { font-size: larger }
h1 { font-size: 150% }
```

PROPIEDAD ABREVIADA DE FUENTE ('FONT')

La propiedad `'font'` es una propiedad resumida para determinar `'font-style'`, `'font-variant'`, `'font-weight'`, `'font-size'`, `'line-height'` y `'font-family'` en una sola propiedad.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'font'	[[<'font-style'> <'font-variant'> <'font-weight'>]? <'font-size'> [/<'line-height'>]? <'font-family'>] caption icon menu message-box small-caption status-bar inherit	Ver propiedades individuales	Todos los elementos.	Si	Ver propiedades individuales	Visual	1
Valor computado:	Ver propiedades individuales						

Todas las propiedades relacionadas con la fuente son reseteadas a sus valores iniciales, y aquellos valores declarados en la propiedad resumida `'font'` reemplazan a sus correspondientes valores iniciales.

Veamos algunos ejemplos de aplicación de la propiedad resumida `'font'`:

```
p { font: 12px/14px sans-serif }
p { font: 80% sans-serif }
p { font: x-large/110% "New Century Schoolbook", serif }
p { font: bold italic large Palatino, serif }
p { font: normal small-caps 120%/120% fantasy }
```

En la segunda regla, el valor del porcentaje del tamaño de la fuente (`'80%'`) se refiere al tamaño de la fuente del elemento padre. En la tercera regla, el porcentaje de la altura de la línea se refiere al tamaño de fuente del

propio elemento En las tres primeras reglas de arriba, `'font-style'`, `'font-variant'` y `'font-weight'` no son mencionadas explícitamente, lo que significa que las tres tomarán sus valores iniciales (`'normal'`). La cuarta regla fija el `'font-weight'` a `'bold'`, el `'font-style'` a `'italic'` e implícitamente fija `'font-variant'` a `'normal'`.

La quinta regla establece `'font-variant'` (`'small-caps'`), `'font-size'` (120% de la fuente del padre), `'line-height'` (120% veces el tamaño de la fuente) y `'font-family'` (`'fantasy'`). Se entiende que la palabra clave `'normal'` se aplica a las dos propiedades restantes: `'font-style'` y `'font-weight'`.

Fuentes del sistema posibles para `'font'`:

Nombre	Explicación
caption	La fuente utilizada para los títulos de los controles (ej., botones, desplegables verticales (drop-downs), etc.).
icon	La fuente utilizada para rotular los iconos.
menú	La fuente utilizada en los menús (menús desplegables verticalmente y listas de menús).
message-box	La fuente utilizada por las cajas de diálogo.
small-caption	La fuente utilizada para rotular controles pequeños.
status-bar	La fuente utilizada en la barra de estado de las ventanas.

Al establecer una fuente de sistema, se establecen todas las propiedades al mismo tiempo (la familia, el tamaño, el peso, el estilo, etc.). Esto las convierte en un tipo resumido que sólo puede ser especificado con la propiedad `'font'` (y no con `'font-family'`).

Vemos el siguiente ejemplo con la propiedad resumida `'font'`:

```
p { font: 400 11px Arial, Tahoma, sans-serif; }
```

Es equivalente a:

```
p {  
  font-family: Arial, Tahoma, sans-serif;  
  font-style: normal;  
  font-variant: normal;  
  font-weight: 400;  
  font-size: 11px;  
  line-height: normal; }
```

14. Texto



A continuación se detallan propiedades que afectan a la presentación visual de caracteres, de espacios, de palabras y de párrafos.

SANGRÍA ('TEXT-INDENT')

Esta propiedad especifica la sangría de la primera línea del texto en un bloque, con respecto al límite izquierdo (o derecho, para las composiciones de derecha a izquierda). Este sangrado debe ser procesado como espacio en blanco por las aplicaciones de usuario.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'text-indent'	<medida> <porcentaje> inherit	0	Elementos a nivel de bloque, celdas de tabla y bloques en línea.	Si	Referido al ancho del bloque de contención.	Visual	1
Valor computado:	El porcentaje según lo especificado o la medida absoluta						

Valores posibles para 'text-indent':

Nombre	Explicación
<medida>	El sangrado es una medida fija
<porcentaje>	El sangrado es un porcentaje del ancho del bloque de contención

El valor de 'text-indent' puede ser negativo, pero puede haber limitaciones específicas de la implementación. Como esta propiedad se hereda, cuando está especificada en un elemento de bloque, afectará a los elementos de bloque en línea descendientes. Por esta razón, es a menudo especificado 'text-indent: 0' en los elementos que son especificados 'display:inline-block'.

Se aplicaría una sangría de 5em a un párrafo de la siguiente manera:

```
p { text-indent: 5em }
```

ALINEACIÓN ('TEXT-ALIGN')

Esta propiedad indica como se alinea el contenido en línea de un bloque, con los valores `'left'`, `'right'`, `'center'` y `'justify'` que equivalen a izquierda, derecha, centrado y justificado respectivamente.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'text-align'	left right center justify inherit	'left' en 'ltr'; 'right' en 'rtl'	Elementos a nivel de bloque, celdas de tabla y bloques en línea.	Si	N/A	Visual	1
Valor computado:	Según el especificado.						

Un bloque de texto es una pila de cajas de línea. Los valores `'left'`, `'right'` y `'center'` especifican cómo las cajas en línea dentro de cada caja de línea se alinean con respecto a los lados izquierdo y derecho de la caja de línea (no en relación con el acceso visual). Para el valor `'justify'`, la AU puede estirar las cajas en línea además de ajustar sus posiciones.

En el siguiente ejemplo, todos los textos estarán alineados a la izquierda, a excepción de los encabezados de nivel 1 que estarán centrados.

```
body { text-align: left; }
h1 { text-align: center; }
```

DECORACIÓN ('TEXT-DECORATION')

Esta propiedad describe las decoraciones (subrayado, tachado, parpadeo, etc.) que se agregan al texto de un elemento usando el color del elemento.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'text-decoration'	none [underline overline line-through blink] inherit	none	Todos los elementos	No	N/A	Visual	1
Valor computado:	Según el especificado.						

Cuando esta propiedad es especificada en un elemento en línea, afecta a todas las cajas generadas por ese elemento. En otros elementos, la propiedad se propaga a una caja en línea anónima que envuelve a todos los hijos en línea en el flujo del elemento y a cualquier descendiente a nivel de bloque en flujo.

Si un elemento no contiene ningún texto, las aplicaciones de usuario no deben procesar las decoraciones (Ej. en elementos que sólo tienen imágenes y espacios en blanco cerrados).

Valores posibles para 'text-decoration':

Nombre	Explicación
none	No produce decoraciones en el texto.

Valores posibles para 'word-spacing':

Nombre	Explicación
normal	El espacio normal entre palabras, según lo definido por la fuente actual y/o la AU.
<medida>	Este valor indica el espacio entre palabras <i>además</i> del espacio entre las palabras por defecto. Los valores pueden ser negativos, pero puede haber límites específicos puestos por la implementación.

En esta ocasión sólo aumentamos el espaciado entre palabras del encabezado **H2**:

```
h2 { word-spacing: 1em }
```

CAPITALIZACIÓN ('TEXT-TRANSFORM')

Esta propiedad controla el efecto de la capitalización (cambios entre mayúsculas y minúsculas) del texto de un elemento.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'text-transform'	capitalize uppercase lowercase none inherit	none	Todos los elementos	Si	N/A	Visual	1
Valor computado:	Según el especificado.						

Valores posibles para 'text-transform':

Nombre	Explicación
capitalize	Pone el primer carácter de cada palabra en mayúscula, sin afectar al resto de los caracteres.
uppercase	Pone todos los caracteres de cada palabra en mayúscula.
lowercase	Pone todos los caracteres de cada palabra en minúsculas.
none	Ningún efecto de capitalización.

De este modo transformamos a mayúsculas todos los elementos h4:

```
h4 { text-transform: uppercase }
```

ESPACIOS EN BLANCO ('WHITE-SPACE')

Esta propiedad declara como son tratados los espacios en blanco dentro del elemento.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'white-space'	normal pre nowrap pre-wrap pre-line inherit	normal	Todos los elementos	Si	N/A	Visual	1
Valor computado:	Según el especificado.						

Valores posibles para 'white-space':

Nombre	Explicación
normal	Indica a las aplicaciones de usuario que deben cerrar las secuencias de espacios en blanco y saltos de líneas para llenar las cajas de línea
pre	Impide a las aplicaciones de usuario cerrar las secuencias de espacios en blanco. Las líneas son cortadas solamente en las nuevas líneas del código fuente, al aparecer "\A" en el contenido generado.
nowrap	Este valor cierra los espacios en blanco como en 'normal', pero suprime los saltos de línea dentro del texto.
pre-wrap	Este valor impide a las aplicaciones de usuario cerrar las secuencias de espacios en blanco. Las líneas son cortadas en las nuevas líneas en el código fuente, o al aparecer "\A" en el contenido generado, y como sea necesario para llenar las cajas de línea
pre-line	Este valor impide a las aplicaciones de usuario cerrar las secuencias de espacios en blanco. Las líneas son cortadas en las nuevas líneas en el código fuente, o al aparecer "\A" en el código generado, y como sea necesario para llenar las cajas de línea.

Veamos algunos ejemplos de aplicación de este atributo sobre diferentes elementos:

```
pre { white-space: pre }  
p { white-space: normal }
```

15. Tablas



INTRODUCCIÓN A LAS TABLAS

Una tabla puede ser utilizada para representar relaciones de tabulaciones entre datos.

Las filas y las columnas de celdas se pueden organizar en grupos de filas y en grupos de columnas. Las filas, las columnas, los grupos de filas, los grupos de columnas, y las celdas pueden tener bordes dibujados alrededor de ellos. Se pueden alinear datos vertical u horizontalmente dentro de una celda y alinear datos en todas las celdas de una fila o de una columna.

En HTML/XHTML la semántica de los distintos elementos de la tabla (**TABLE**, **CAPTION**, **THEAD**, **TBODY**, **TFOOT**, **COL**, **COLGROUP**, **TH**, y **TD**) está bien definida, sin embargo, en otros lenguajes de documento (tales como aplicaciones XML) pueden no existir elementos de tabla predefinidos. Por ello, CSS permite hacer pasar cualquier elemento como un elemento de tabla a través de la propiedad `'display'`. Por ejemplo, la siguiente regla hace que el elemento FOO actúe como un elemento **TABLE** de HTML y el elemento BAR actúe como un elemento **CAPTION**:

```
FOO { display : table }  
BAR { display : table-caption }
```

El término **elemento de la tabla** se refiere a cualquier elemento implicado en la creación de una tabla. Un elemento de tabla "interno" es uno que produce una fila, un grupo de filas, una columna, un grupo de columnas o una celda.

EL MODELO DE TABLA DE CSS

El modelo de tablas de CSS se basa en el modelo de tablas de HTML, donde una tabla consiste en un encabezado opcional y cualquier número de filas de celdas. Este es un modelo "principalmente de filas" porque se especifican filas y no columnas, las cuáles son derivadas una vez que las filas son especificadas (la primera celda de cada fila pertenece

a la primera columna, la segunda a la segunda columna, etc.).

Así, el modelo de tabla consiste en tablas, encabezados, filas, grupos de filas, columnas, grupos de columnas y celdas. El modelo de CSS no requiere que el lenguaje del documento incluya los elementos que corresponden a cada uno de estos componentes. Como ya indicamos, para los lenguajes de documento que no tienen elementos de tabla predefinidos se debe convertir elementos, a través de la propiedad `'display'`, en elementos de tabla.

Los valores para 'display' en relación con las tablas:

En CSS	En HTML:	Explicación
table	TABLE	Especifica que un elemento define una tabla a nivel de bloque: es un bloque rectangular que participa de un contexto de formato de bloque.
inline-table	TABLE	Especifica que un elemento define una tabla a nivel de línea: es un bloque rectangular que participa de un contexto de formato en línea.
table-row	TR	Especifica que un elemento es una fila de celdas.
table-row-group	TBODY	Especifica que un elemento agrupa una o más filas.
table-header-group	THEAD	Especifica que un elemento agrupa una o más filas, que son mostradas antes que el resto de filas y de grupos de filas y después de cualquier encabezado superior. Las aplicaciones de usuario que imprimen pueden repetir filas de encabezado en cada página ocupada por una tabla.
table-footer-group	TFOOT	Especifica que un elemento agrupa una o más filas, que son mostradas después del resto de filas y de grupos de filas y antes de cualquier encabezado inferior. Las aplicaciones de usuarios que imprimen pueden repetir filas de pie en cada página ocupada por una tabla.
table-column	COL	Especifica que un elemento describe una columna de celdas.
table-column-group	COLGROUP	Especifica que un elemento agrupa una o más columnas.
table-cell	TD, TH	Especifica que un elemento representa una celda de la tabla.
table-caption	CAPTION	Especifica un encabezado para la tabla. Sólo debe ponerse un elemento con <code>'display:caption'</code> dentro de un elemento <code>table</code> o <code>inline-table</code> .

COLUMNAS

Las celdas de una tabla pueden pertenecer al contexto de filas o columnas, sin embargo, en el código fuente las celdas son descendientes de las filas, nunca de las columnas. No obstante, algunos aspectos de las celdas pueden ser influenciados poniendo propiedades a las columnas.

Propiedades de columnas y grupos de columnas:

Nombre	Explicación
'border'	El borde se aplica a las columnas solo si <code>'border-collapse'</code> es definido como <code>'collapse'</code> sobre el elemento tabla. En ese caso, los bordes definidos para las columnas y grupos de columnas son introducidos al algoritmo de resolución de conflictos que elige el estilo del borde para cada lado del borde de la celda.
'background'	Determina el fondo para las celdas en la columna, pero sólo si la celda y la fila tienen fondos transparentes.
'width'	Determina el ancho mínimo de la columna.

Nombre	Explicación
'visibility'	Si el valor de esta propiedad es 'collapse', ninguna de las celdas en la columna son procesadas, y las celdas contenidas dentro de otras columnas son recortadas. Además, el ancho de la tabla es disminuido por el ancho de la columna que se hubiera quitado. Ver "efectos dinámicos" abajo. Otros valores para 'visibility' no tienen efecto.

Las primeras dos reglas juntas implementan el atributo "rules" de HTML 4.0 con un valor de "cols". La tercera regla hace que la columna "totales" sea azul y las dos reglas finales muestran como hacer una columna de tamaño fijo, usando el algoritmo de estructura fija.

```
col { border-style: none solid }
table { border-style: hidden }
col.totales { background: blue }
table { table-layout: fixed }
col.totales { width: 5em }
```

LAS TABLAS EN EL MODELO DE FORMATO VISUAL

En el modelo de formato visual, una tabla debe funcionar como un elemento a nivel de bloque o a nivel de línea. Las tablas tienen contenido, relleno, bordes y márgenes.

En ambos casos, los elementos de la tabla generan una caja anónima que contiene a la propia caja de la tabla y a la caja del encabezado (si está presente). Las cajas de tabla y de encabezados retienen sus propias áreas de contenido, relleno, márgenes y bordes, y las dimensiones de la caja rectangular anónima son las mínimas requeridas para contener a ambas. Los márgenes verticales se cierran donde la caja de la tabla y la caja del encabezado se tocan. Cualquier reposicionamiento de la tabla debe mover la caja anónima entera, no solo la caja de la tabla, de modo que el encabezado siga a la tabla.

Posición y alineación del encabezado

La propiedad 'caption-side' especifica la posición de la caja del encabezado con respecto a la caja de la tabla.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'caption-side'	top bottom inherit	top	Elementos 'table-caption'	Si	N/A	Visual	1
Valor computado:	Según el especificado.						

Valores posibles para 'caption-side':

Nombre	Explicación
top	Posiciona la caja del encabezado encima de la caja de la tabla.
bottom	Posiciona la caja del encabezado abajo de la caja de la tabla.

Los encabezados encima o debajo del elemento 'table' son estructurados como si fueran un elemento de bloque antes y después de la tabla, excepto que (1) hereden las propiedades heredables de la tabla, y (2) no se consideran como una caja de bloque para el propósito de ningún elemento 'run-in' que pueda preceder a la tabla. Un encabezado que está arriba o abajo de una caja de tabla también se comporta como una caja de bloque para el cálculo del ancho y el alto; el ancho y alto son calculados con respecto al bloque de contención de la caja de la tabla.

Para alinear el contenido del encabezado horizontalmente dentro de la caja del encabezado, se utiliza la propiedad 'text-align'.

En el siguiente ejemplo, el encabezado será colocado debajo de la tabla con el mismo ancho que ésta y el

texto centrado:

```
caption {
caption-side: bottom;
width: auto;
text-align: center; }
```

COMPOSICIÓN VISUAL DEL CONTENIDO DE LAS TABLAS

Los elementos internos de una tabla generan cajas rectangulares con contenido, bordes y relleno, pero sin márgenes. La composición visual de estas cajas está gobernada por una rejilla rectangular, irregular de filas y columnas.

Capas y transparencia en la tabla

Para encontrar el fondo de cada celda de la tabla, los distintos elementos de la tabla pueden considerarse como si estuvieran en seis capas superpuestas. El fondo aplicado sobre un elemento en una de estas capas solo será visible si las capas encima de ella tienen un fondo transparente.

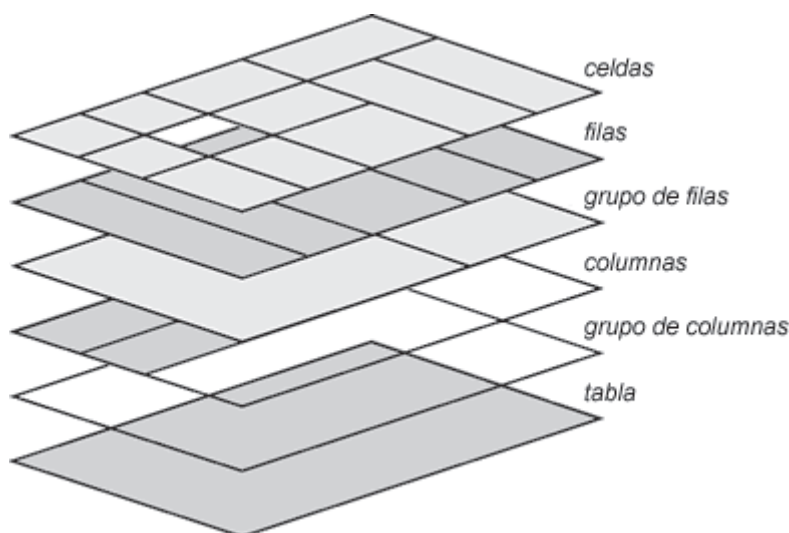


Tabla: La capa inferior es un plano que representa a la caja de la tabla en sí (puede ser transparente).

Grupo de columnas: Es la capa siguiente, donde cada grupo de columnas se extiende desde la parte superior de las celdas en la fila superior hasta la inferior de las celdas en la fila inferior y desde el borde izquierdo de la columna más a la izquierda hasta el borde derecho de la columna más a la derecha. El fondo se extiende hasta cubrir toda el área de todas las celdas que provienen del grupo de columnas, pero esta extensión no afecta al posicionado de la imagen de fondo.

Columnas: Cada columna es tan alta como el grupo de columnas y tan amplia como una celda normal en la columna. El fondo se extiende hasta cubrir toda el área de todas las celdas que provienen de la columna, incluso si su contenido sobresale de la columna, pero esta extensión no afecta al posicionamiento de la imagen de fondo.

Grupo de filas: La siguiente es la capa que contiene los grupos de filas, donde cada uno se extiende desde la esquina superior izquierda de su celda más arriba en la primera columna hasta el borde inferior derecho de su celda más abajo en la última columna.

Filas: La penúltima capa contiene las filas. Cada fila es tan amplia como el grupo de filas y tan alta como una celda normal en el flujo. Como con las columnas, el fondo se extiende hasta cubrir toda el área de todas las celdas que son

originadas en la fila, incluso si su contenido sobresale de la fila, pero esta extensión no afecta al posicionamiento de la imagen de fondo.

Celdas: son contenidos en la última capa y aunque todas las filas contengan el mismo número de celdas, puede que no todas las celdas tengan especificado un contenido. Si el valor de su propiedad `'empty-cells'` es `'hide'` estas celdas "vacías" son transparentes dejando ver el fondo de la celda, la fila, el grupo de filas, la columna y el grupo de columnas, mostrando el fondo de la tabla.

Los límites de las filas, columnas, grupos de filas y grupos de columnas en el modelo de bordes cerrados coincide con las hipotéticas líneas de la rejilla en la que los bordes de las celdas son centradas (en este modelo, las filas y columnas juntas cubren exactamente la tabla sin dejar huecos). En el modelo de bordes separados, el límite coincide con el límite del borde de las celdas (puede haber huecos entre las filas, las columnas, los grupos de filas o los grupos de columnas, correspondiendo a la propiedad `'border-spacing'`).

Si la tabla tiene `'border-collapse: separate'`, el fondo del área dada por la propiedad `'border-spacing'` es siempre el fondo del elemento **TABLE** (Ver ["Modelo de bordes separados"](#)).

Algoritmos para el ancho de la tabla ('table-layout')

Si los márgenes de la tabla son fijados a `'0'` y el ancho a `'auto'`, la tabla no pondrá el tamaño automáticamente para llenar su bloque de contención. Las tablas pueden ser centradas usando los márgenes izquierda y derecha como `'auto'`.

La propiedad `'table-layout'` controla el algoritmo utilizado para componer las celdas, filas y columnas de la tabla.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'table-layout'	auto fixed inherit	auto	a: Elementos 'table' e 'inline-table'	No	N/A	Visual	1
Valor computado:	Según el especificado.						

Valores posibles para **'table-layout'**:

Nombre	Explicación
fixed	Se utiliza el algoritmo de composición fija de la tabla.
auto	Se utiliza cualquier algoritmo de composición de tablas.

Composición fija de la tabla

Con este algoritmo, la composición horizontal de la tabla no depende del contenido de las celdas, sino del ancho de la tabla, el ancho de las columnas y los bordes o espacio entre las celdas.

El ancho de la tabla puede ser especificado con la propiedad `'width'`. Un valor de `'auto'` (tanto para `'display: table'` como para `'display: inline-table'`) significa que se utiliza el algoritmo de composición automática de la tabla.

Si la AU soporta composición fija de tabla cuando `'width'` es `'auto'`, lo siguiente creará una tabla que es 4em más estrecha que el bloque de contención:

```
table { table-layout: fixed;
margin-left: 2em;
margin-right: 2em }
```

En el algoritmo de composición fija de la tabla, el ancho de cada columna es determinado como sigue:

1. Un elemento columna con un valor distinto a `'auto'` para la propiedad `'width'` determina el ancho de

esa columna.

2. Si no, una celda en la primera fila con un valor distinto de `'auto'` para la propiedad `'width'` determina el ancho para esa columna. Si la celda abarca más de una columna, el ancho se divide entre las columnas.
3. Las columnas restantes se dividen el espacio horizontal restante de la tabla (menos los borde o el espacio entre las celdas).

El ancho de la tabla es entonces el mayor de los valor de la propiedad `'width'` para el elemento tabla y la suma de los anchos de las columnas (más el espacio entre celdas o bordes). Si la tabla es mas amplia que las columnas, el espacio extra debe ser distribuido entre las columnas.

Composición automática de la tabla

En este algoritmo, el ancho de la tabla está dado por el ancho de sus columnas (y los bordes que intervienen).

Este algoritmo puede resultar ineficiente dado que requiere que las aplicaciones de usuario tengan acceso a todo el contenido de la tabla antes de determinar la composición final y puede demandar más de una pasada.

El ancho de las columnas se determina como sigue:

1. Se calcula el ancho mínimo del contenido (AMC) de cada celda: el contenido estructurado puede abarcar cualquier número de líneas pero no puede desbordar la caja de la celda. Si el 'ancho' (A) especificado de la celda es mayor que AMC, A es el ancho mínimo de la celda. Un valor `'auto'` significa que AMC es el ancho mínimo de la celda. Además, se calcula el ancho "máximo" de celda de cada celda: se da formato al contenido sin saltos de línea salvo donde aparecen los saltos de línea explícitos.
2. Por cada columna, se determina un ancho máximo y mínimo de columna para las celdas que ocupan solo esa columna. El mínimo es el requerido por la celda con el mayor ancho mínimo de celda (o el 'ancho' de la columna, cualquiera que sea mayor). El máximo es el requerido por la celda con mayor ancho máximo de celda (o el 'ancho' de la columna, cualquiera que sea mayor).
3. Por cada celda que abarca más de una columna, se incrementan los anchos mínimos de las columnas que abarca de modo que juntas, estas al menos tan anchas como la celda. Se hace lo mismo para los anchos máximo. Si es posible, se ensancha las columnas abarcadas en aproximadamente el mismo valor.

Esto da un ancho máximo y mínimo para cada columna. Los anchos de las columnas influyen en el ancho final de la tabla como sigue:

1. Si la propiedad `'width'` del elemento `'table'` o `'inline-table'` tiene un valor computado (A) distinto de `'auto'`, el valor de la propiedad utilizado para la composición es el mayor entre A y el ancho mínimo requerido por todas las columnas más el espacio entre las celdas o bordes (MIN). Si A es mayor que MIN, el ancho extra deberá ser distribuido entre las columnas.
2. Si el elemento `'table'` o `'inline-table'` tiene `'width: auto'`, el ancho de la tabla utilizado para la composición es el mayor entre el ancho del bloque de contención de las tablas y MIN. Sin embargo, si el ancho máximo requerido por las columnas más el espacio entre las celdas o bordes (MAX) es menor que el bloque de contención, se usa MAX. Un valor de porcentaje para un ancho de columna es relativo al ancho de la tabla. Si la tabla tiene `'width: auto'`, un porcentaje representa una condición en el ancho de la columna, la que una UA deberá tratar de satisfacer. (Obviamente, esto no siempre es posible: si el ancho de la columna es `'110%'`, la condición no puede satisfacerse.)

Algoritmo para la altura de la tabla

La altura de la tabla está dada por la propiedad `'height'` del elemento `'table'` o `'inline-table'`. Un valor de `'auto'` significa que la altura es la suma de las alturas de las filas mas cualquier espacio entre celdas o bordes. Cualquier otro valor especifica la altura explícitamente; la tabla puede resultar de ese modo más larga o más corta que la altura de sus filas. CSS2.1 no especifica un procesamiento cuando la altura especificada de la tabla difiere de la altura del contenido, en particular si la altura del contenido debe sustituir la altura especificada; si no, como debe distribuirse el espacio extra entre las filas que sumadas resultan menores que la altura especificada para la tabla; o, si la altura del

contenido excede la altura especificada de la tabla, si la AU debe proporcionar un mecanismo de desplazamiento.
Nota: Futuras versiones de CSS pueden especificar esto mejor.

La altura de la caja del elemento `'table-row'` se calcula una vez que la aplicación del usuario tiene disponible todas las celdas de la fila: es la máxima `'altura'` especificada para la fila y la mínima altura (MIN) requerida por las celdas. Un valor de `'height'` como `'auto'` para un `'table-row'` significa que la altura de la fila utilizada para la composición es MIN. MIN depende de la caja de la celda y de la alineación de la caja de la celda de (muy parecido al cálculo de la altura de una caja de línea). CSS2.1 no define a que se refieren los valores de porcentajes en `'height'` cuando se especifican para las filas o grupos de filas de la tabla.

En CSS2.1, la altura de una caja de celda es el máximo de la propiedad `'height'` de las celdas de la tabla y la mínima altura requerida por el contenido (MIN). Un valor `'auto'` para `'height'` implica que el valor MIN será usado para la composición. CSS2.1 no especifica a que se refieren los valores de porcentaje en `'height'` cuando se especifican para las celdas de la tabla.

CSS2.1 no especifica como las celdas que abarcan más de una fila afectan al cálculo de la altura de la fila excepto que la suma de las alturas de las filas involucradas debe ser suficiente para contener a la celda que abarca las filas.

La propiedad `'vertical-align'` de cada celda determina su alineación dentro de la fila. Cada contenido de cada celda tienen una línea base, una parte superior, una parte media y una parte inferior, como la propia fila. En el contexto de las tablas, los valores para `'vertical-align'` tienen los siguientes significados:

- **baseline:** La línea base de la celda se pone a la misma altura que la línea base de las primeras filas que abarca (ver abajo para la definición de líneas de base de celda y filas).
- **top:** La parte superior de la caja es alineada con la parte superior de la primera fila que contiene.
- **bottom:** La parte inferior de cada caja de la celda es alineada con la parte inferior de la última fila que contiene.
- **middle:** El centro de la celda es alineado con el centro de las filas que contiene.
- **sub, super, text-top, text-bottom, <medida>, <porcentaje>:** Estos valores no se aplican a las celdas, en cambio la celda se alinea por la línea de base.

Alineación horizontal en una columna

La alineación horizontal del contenido de una celda dentro de una caja de celda se especifica con la propiedad `'text-align'`.

Efectos dinámicos en filas y columnas

La propiedad `'visibility'` toma el valor `'collapse'` para filas, grupos de filas, columnas y grupos de columnas. Este valor provoca que la fila o columna entera se deje de mostrar, y el espacio normalmente ocupado por la fila o columna quede disponible para otro contenido. El contenido de las filas y columnas cerradas (`span`) que se cruzan con columnas y filas colapsadas, es recortado. La supresión de la fila o columna, sin embargo, no afecta de ningún modo a la composición de la tabla. Esto permite efectos dinámicos al quitar filas de tablas o columnas sin forzar una recomposición de la tabla para calcular el cambio potencial en los requerimientos de la columna.

BORDES

Hay dos modelos distintos para definir los bordes de las celdas de una tabla en CSS. Uno es más conveniente para los llamados bordes separados alrededor de las celdas individuales, el otro es conveniente para bordes continuos de una punta a otra de la tabla.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'border-collapse'	collapse separate inherit	separate	Elementos 'table' y 'inline-table'	Si	N/A	Visual	1
Valor computado:	Según el especificado.						

Esta propiedad selecciona un modelo de borde de la tabla. El valor 'separate' selecciona el modelo de bordes separado. El valor 'collapse' selecciona el modelo de bordes cerrado.

El modelo de bordes separado

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'border-spacing'	<medida> <medida>? inherit	0	Elementos 'table' y 'inline-table'*	Si	N/A	Visual	1
Valor computado:	Las dos medidas absolutas						

* **Nota:** Las aplicaciones de usuario pueden también aplicar la propiedad 'border-spacing' a elementos 'frameset' como un valor sustituto para el atributo no estándar 'framespacing'.

Las medidas especifican la distancia que separa los bordes de las celdas adyacentes. Si se especifica una sola medida, esta da el espacio horizontal y vertical. Si se especifican dos medidas, la primera da el espacio horizontal y la segunda el espacio vertical. Las medidas no pueden ser negativas.

La distancia entre el borde de la tabla y los bordes de las celdas en el límite de la tabla es el relleno para ese lado, más la distancia de espacio del borde relevante. Por ejemplo, en el lado derecho, la distancia del relleno derecho más el espacio del borde horizontal.

En este modelo, cada celda tiene un borde individual. La propiedad 'border-spacing' especifica la distancia entre los bordes de celdas adyacentes. En este espacio, el fondo de la fila, la columna, el grupo de filas, y el grupo de columnas es visible, permitiendo mostrar a través del mismo el fondo de la tabla. Las filas, columnas, grupos de filas y grupos de columnas no pueden tener bordes.

Bordes y fondos alrededor de celdas vacías ('empty-cells')

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'empty-cells'	show hide inherit	show	Elementos 'table-cell'	Si	N/A	Visual	1
Valor computado:	Según el especificado.						

En el método de bordes separados, esta propiedad controla el procesamiento de bordes y fondos alrededor de las celdas que no tienen contenido visible. Las celdas vacías con la propiedad 'visibility' fijadas como 'hidden' se consideran que no tienen un contenido visible. El contenido visible incluye " " y otro espacio en blanco excepto ASCII CR ("    "), LF ("    "), tab ("    "), y espacio ("    ").

Valores posibles para 'empty-cells':

Nombre	Explicaci��n
show	Los bordes y fondos son dibujados alrededor/detr��s de las celdas vac��as (como celdas normales).

Nombre	Explicación
hide	Los bordes o fondos no se dibujan alrededor/detrás de las celdas vacías. Si todas las celdas de una fila tienen el valor <code>'hide'</code> y no tienen contenido, la fila entera se comporta como si tuviera <code>'display: none'</code> .

Con la siguiente regla se dibujan el borde y fondo en todas las celdas (incluso las vacías):

```
table { empty-cells: show }1
```

El modelo de bordes cerrados

En el modelo de bordes cerrados, es posible especificar los bordes que rodean a toda o parte de una celda, fila, grupo de filas, columna y grupo de columnas. Los bordes para los atributos `"rules"` de HTML pueden ser especificados de este modo.

Los bordes son centrados con las líneas de la rejilla entre las celdas.

Las AUs deben calcular un ancho inicial de borde izquierdo y derecho para la tabla mediante el examen de la primera y última celda en la primera fila de la tabla. El ancho del borde izquierdo de la tabla es la mitad del borde derecho de la última celda cerrada. Si la secuencia de filas tiene muchos bordes derechos e izquierdos cerrados, entonces cualquier exceso se expande dentro del área del margen de la tabla.






El ancho del borde superior de la tabla es calculado examinando todas las celdas cuyos bordes superiores se cierran con el borde superior de la tabla. El ancho del borde de la tabla es igual a la mitad del máximo borde superior cerrado. El borde inferior es calculado examinando todas las celdas cuyo borde inferior se cierra con el inferior de la tabla. El ancho del borde inferior es igual a la mitad del máximo borde inferior cerrado.




Cualquier borde que se extiende dentro del margen se tiene en cuenta cuando se determina si la tabla desborda algún ascendente.

En este modelo, el ancho de la tabla incluye la mitad del borde de la tabla. Además, una tabla no tiene relleno (pero tiene márgenes).

Estilos de bordes

Algunos de los valores de `'border-style'` tienen diferentes significados para las tablas que para otros elementos. En la lista de abajo están marcados con un asterisco.

Nombre	Explicación	
none	ningún borde; el ancho del borde es cero	
hidden *	Igual que <code>'none'</code> , pero en el modelo de bordes cerrados, también inhibe cualquier otro borde (ver la sección sobre conflictos de bordes).	
dotted	El borde es una serie de puntos.	
dashed	El borde es una serie de pequeños segmentos de líneas.	
solid	El borde es un único segmento de línea.	
double	El borde son dos líneas sólidas. La suma de las dos líneas y el espacio entre ellas es igual al valor de <code>'border-width'</code> .	
groove	El borde luce como si estuviera tallado en el lienzo.	

Nombre	Explicación	
ridge	Lo opuesto a ' groove ': el borde parece que estuviera sobresaliendo (en relieve) del lienzo.	
inset *	En el modelo de bordes separados, el borde provoca que toda la caja luzca como si estuviera embebida (incrustada) en el lienzo. En el modelo de bordes cerrados, es igual a ' ridge '.	
outset *	En el modelo de bordes separados, el borde provoca que toda la caja luzca como si estuviera en relieve (sobresaliendo) sobre el lienzo. En el modelo de bordes cerrados, es igual a ' groove '.	




16. Interfaz de usuario












CURSORES ('CURSOR')

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'cursor'	[[<uri> ,]* [auto crosshair default pointer move e-resize noresize nw-resize n-resize se-resize sw-resize s-resize w-resize text wait help progress]] inherit	auto	Todos los elementos.	Si	N/A	Visual, interactivos	2
Valor computado:		URI absoluta, sino como el especificado					

Valores posibles para 'cursor' (especifica el tipo de cursor que se exhibirá para el dispositivo señalizador):

Nombre	Explicación
auto	La AU determina el cursor a mostrar basado en el contexto actual.
crosshair	 Una cruz simple (ej., segmentos cortos de línea semejantes a un signo "+")
default	 El cursor predeterminado dependiente de la plataforma. Procesado a menudo como una flecha.
pointer	 El cursor es un puntero que indica un enlace.

Nombre		Explicación
move		Indica que algo será movido.
e-resize, w-resize,		Indique que algún borde será movido. Por ejemplo, el cursor ' se-resize ' se usa cuando el movimiento comienza desde la esquina sudeste de la caja.
n-resize, s-resize,		
ne-resize, sw-resize,		
nw-resize, se-resize,		
text		Indica el texto que puede seleccionarse. Procesado a menudo como barra-I
wait		Indica que el programa está ocupado y el usuario debe esperar. Procesado a menudo como un reloj de pulsera o un reloj de arena.
progress		Un indicador de progreso. El programa está realizando algún proceso, pero es diferente de ' wait ' en el que el usuario todavía puede actuar recíprocamente con el programa. Procesado a menudo como una pelota de playa girando, o una flecha con un reloj o reloj de arena.
help		La ayuda está disponible sobre el objeto señalado por el cursor. Procesado a menudo como un signo de interrogación o un globo.
<uri>		La aplicación de usuario recupera el cursor del recurso señalado por el URI. Si la aplicación del usuario no puede manejar el primer cursor de una lista de cursores, debe procurar manejar el segundo, etc. Si la aplicación de usuario no puede manejar ningún cursor definido por el usuario, debe utilizar el cursor genérico al final de la lista.

Un ejemplo de aplicación de cursores podría ser el siguiente:

```
:link { cursor: url(icono.gif), pointer }
```

En él se fija el cursor para que en los hipervínculos sin visitar sean mostrados con una imagen determinada, que en caso de no ser soportada deberá ser reemplazada por el siguiente valor que es '**pointer**'.

En cualquier sitio navegable, normalmente se verá el cursor '**default**' que cambiará por '**text**' sobre cualquier texto y por '**pointer**' sobre cualquier hipervínculo.

🔗 [Verificar diferentes tipos de cursores \(CD > ejemplos > css > interfaz > cursors_01.html\)](#)

COLORES DEL SISTEMA CSS2

Nota: Los colores del sistema CSS2 son desaprobados en el módulo de color de CSS3.

Tal como vimos en el capítulo de [Valores](#), además de poder asignar valores de color predefinidos, se pueden introducir un sistema de valores de nombres de color que permite a especificar colores de manera que se integren en el entorno gráfico del sistema operativo.

Para los sistemas que no tienen un valor correspondiente, el valor especificado se debe aproximar al valor más cercano del sistema, o a un color por defecto.

Cualquier propiedad de color (ej., '**color**' o '**background-color**') puede tomar uno de los nombres de este sistema y aunque éstos son sensibles a mayúsculas y minúsculas, se recomienda que la mezcla de mayúsculas y minúsculas mostrada sea utilizada, para hacer los nombres más legibles.

🔗 Para ver los nombres de colores de sistema, su explicación y apariencia vea la tabla de colores del sistema (CD ejemplos > css > valores > colores.html#colores_sistema)

Se puede colocar como color de texto y de fondo los mismos que se utilizan en la ventana del usuario, de la siguiente manera:

```
p { color: WindowText; background-color: Window }
```

PREFERENCIAS DE FUENTES DEL USUARIO

De igual modo que para los colores, los autores pueden especificar las fuentes de manera que hagan uso de los recursos del sistema de un usuario.

CONTORNOS DINÁMICOS ('OUTLINE')

Las propiedades del contorno controlan el estilo de los contornos dinámicos, en objetos visuales tales como botones, campos activos de formularios, mapas de imágenes, etc., para enfatizarlos. Los contornos se diferencian de los bordes en los siguientes sentidos:

- no ocupan espacio.
- pueden ser no rectangulares.

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda?	Porcentajes	Medio	W3C
'outline-color'	<color> invert inherit	invert	Todos	No	N/A	visual, interactivo	2
'outline-style'	<border-style> inherit	none					
'outline-width'	<border-width> inherit	medium					
'outline'	['outline-color' 'outline-style' 'outline-width'] inherit	Ver propiedades individuales					
Valor computado:	'outline': Ver propiedades individuales 'outline-width': Medida absoluta; '0' si el estilo outline es 'none' o 'hidden' 'outline-style' y 'outline-color': Según el especificado						

El contorno creado con las propiedades de contorno es dibujado "sobre" una caja (siempre encima) y no influye en la posición o el tamaño de la caja, o de ninguna otra caja. Por lo tanto, mostrar u ocultar contornos no hace que el objeto debe dibujarse nuevamente.

El contorno puede ser dibujado comenzando justo fuera del límite del borde.

Los contornos pueden ser no-rectangulares. Por ejemplo, si el elemento está dividido en varias líneas, el contorno es el contorno mínimo que encierra a todas las cajas del elemento. En contraste con los bordes, el contorno no está abierto en la línea de finalización o comienzo de la caja, pero está siempre totalmente conectado si es posible.

La propiedad 'outline-width' acepta los mismos valores que 'border-width'. La propiedad 'outline-style' acepta los mismos valores que 'border-style', excepto que 'hidden' no es un valor lícito. La propiedad 'outline-color' acepta todos los colores, como así también la palabra clave 'invert'.

'**invert**' debe producir una inversión de color en los píxeles de la pantalla, lo que asegura que el borde del foco sea visible, sin que importe el color del fondo.

La propiedad '**outline**' es una propiedad resumida, y determina al mismo tiempo '**outline-style**', '**outline-width**', y '**outline-color**'. A diferencia de los bordes no se puede asignar un contorno diferente para cada lado.

A continuación vemos un contorno para un botón:

```
button { outline : thick solid blue; }
```

🔗 [Verificar el efecto del contorno \(CD > ejemplos > css > interfaz > outlines.html\)](#)

Contornos y foco

Las interfaces gráficas pueden utilizar contornos alrededor de elementos para indicar al usuario qué elemento de la página tiene **el foco**, es decir que se produce una interacción del usuario con el documento (ej., introduciendo texto, seleccionando un botón, etc.). Estos contornos son sumados a cualquier borde y pasan de contornos activados a desactivados sin dibujar nuevamente el documento. Las aplicaciones de usuario que dan soporte a los grupos de medios interactivos deben mantener identificada la ubicación del foco y representar el foco, lo que puede realizarse utilizando contornos dinámicos conjuntamente con la pseudo-clase :focus.

Pueden utilizarse las siguientes reglas para dibujar una línea negra gruesa alrededor de un elemento cuando tiene el foco, y una línea roja gruesa cuando es activo:

```
:focus { outline: thick solid black }  
:active { outline: thick solid red }
```


Anexos



INFORMACIÓN COMPLEMENTARIA

1. Información de referencia de SGML para XHTML 1.0



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.

- **DTD XHTML 1.0 por defecto estricto** (Ver: CD > Anexos > xhtml1-strict.dtd)
- **DTD XHTML 1.0 Transicional** (Ver: CD > Anexos > xhtml1-transitional.dtd)
- **DTD XHTML 1.0 con marcos** (Ver: CD > Anexos > xhtml1-frameset.dtd)

VALIDACIÓN DE DOCUMENTOS

Muchos autores se basan en un conjunto limitado de navegadores para comprobar los documentos que producen, y suponen que si los navegadores pueden representar sus documentos, es que éstos son válidos. Desafortunadamente, esto es una manera muy poco efectiva de verificar la validez de un documento, precisamente porque los navegadores están diseñados para que al encontrar documentos no válidos los represente lo mejor posible para evitar la frustración de los usuarios.

Tenga en cuenta que esta validación, aunque es útil y muy recomendable, no garantiza que el documento sea completamente conforme con la especificación XHTML, sólo asegura que la sintaxis, la estructura, la lista de elementos y sus atributos son válidos. De todos modos, este tipo de validación sigue siendo altamente recomendable, ya que permite la detección de errores que hacen a los documentos no válidos.

ENTIDADES DE CARACTERES

Introducción a las referencias a entidades de caracteres

Una referencia a una entidad de caracteres es una estructura SGML que hace referencia a un carácter del conjunto de caracteres del documento.

Esta versión de HTML soporta varios conjuntos de referencias a entidades de caracteres:

- **Caracteres de la ISO 8859-1 o Latin-1** (Ver: CD > Anexos > xhtml1-lat1.ent)
- **Símbolos, símbolos matemáticos y letras griegas** (Ver: CD > Anexos > xhtml1-symbol.ent)
- **Caracteres con significado en el código y de internacionalización** (Ver: CD > Anexos > xhtml1-special)

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 (€ o € o €) se define como una parte de los caracteres especiales.

Referencias a entidades de caracteres para caracteres de la ISO 8859-1

Las referencias a entidades de caracteres de esta sección producen caracteres cuyos equivalentes numéricos ya deberían estar soportados por los agentes de usuario conformes con HTML 2.0. Así, la referencia a entidad de caracteres ÷ es una forma de obtener el signo de división (÷) más conveniente que ÷. Para soportar estas entidades con nombre, los agentes de usuario no tienen más que reconocer los nombres de las entidades y convertirlos a caracteres que caigan dentro del repertorio de la [ISO88591]. (Ver: CD > Anexos > xhtml1-lat1.ent)

Referencias a entidades de caracteres para símbolos, símbolos matemáticos y letras griegas

Las referencias a entidades de caracteres de esta sección producen caracteres que pueden ser representados por los signos de la fuente de Adobe Symbol, ampliamente extendida, que incluye caracteres griegos, varios símbolos de agrupamiento, y una selección de operadores matemáticos como símbolos de gradiente, producto y suma. (Ver: CD > Anexos > xhtml1-symbol.ent)

Cuándo usar entidades griegas. Este conjunto de entidades contiene todas las letras usadas en el griego moderno. Sin embargo, no contiene la puntuación griega, caracteres acentuados precompuestos ni los acentos sin espaciado (tonos, dialytika) necesarios para componerlos. No hay letras arcaicas, letras exclusivas del copto, ni letras precompuestas para el griego politónico. Las entidades aquí definidas no están pensadas para la representación de textos en griego moderno y como tales no serían una buena representación; están pensadas, más bien, para las letras griegas ocasionales que aparecen en obras técnicas y matemáticas.

Referencias a entidades de caracteres para caracteres con significado en el código y para caracteres de internacionalización

Las referencias a entidades de caracteres de esta sección son para convertir caracteres significativos para el código en secuencias de escape, y para denotar espacios y guiones. Otros caracteres de esta sección se aplican a cuestiones de internacionalización como la eliminación de las ambigüedades del texto bidireccional. (Ver: CD > Anexos > xhtml1-special)

2. Direccionalidad



INTRODUCCIÓN AL ALGORITMO BIDIRECCIONAL

Con la intervención de dos idiomas, el inglés (de izquierda a derecha), y el hebreo (de derecha a izquierda), podríamos desarrollar un ejemplo para el algoritmo bidireccional, tal como sigue:

```
inglés1 HEBREO2 inglés3 HEBREO4 inglés5 HEBREO6
```

En el ejemplo el primer carácter de la fila es "i", el segundo es "n", y el último es "6".

Si el idioma del documento que contiene a este párrafo es el inglés (izquierda a derecha), el mismo sería correctamente presentado de la siguiente forma:

```
inglés1 2OERBEH inglés3 4OERBEH inglés5 6OERBEH
<-----<-----<-----
      H          H          H
----->
                        I
```

El inglés predomina y hay algunos textos incluidos en hebreo. En cambio, si el idioma predominante del documento es el hebreo, la dirección base es de derecha a izquierda y la presentación correcta sería:

```
6OERBEH inglés5 4OERBEH inglés3 2OERBEH inglés1
----->----->----->
      I          I          I
<----->
                        H
```

Así, la frase entera se escribe de derecha a izquierda, con las secuencias en inglés correctamente invertidas con el algoritmo bidireccional.

HERENCIA DE LA INFORMACIÓN SOBRE LA DIRECCIÓN DEL TEXTO

El algoritmo bidireccional obliga a que los bloques de texto tengan una dirección base, la cuál debe ser especificada con el atributo `dir` de dicho elemento. El valor por defecto de este atributo es de izquierda a derecha (ltr).

El atributo `dir` aplicado a un elemento en bloque, tiene efecto hasta la finalización del bloque y en cualquier elemento en bloque anidado. Si se establece el atributo `dir` para algún elemento anidado, se anula en él el valor heredado.

La dirección base de un documento se especifica estableciendo el atributo `dir` del elemento `html`, tal como sigue:

```
<html dir="RTL">
```

Los elementos en línea no heredan el atributo `dir`, por lo cuál un elemento en línea sin un atributo `dir` no abre un nivel adicional de inclusión con respecto al algoritmo bidireccional.

ESPECIFICACIÓN DE LA DIRECCIÓN DEL TEXTO INCLUIDO

Tal como se mostró en los ejemplos anteriores, el algoritmo bidireccional invierte las secuencias incluidas de caracteres de acuerdo con su direccionalidad inherente, sin embargo, en general sólo se puede tener en cuenta un nivel de inclusión. Para lograr niveles adicionales de cambios de dirección debe hacerse uso del atributo `dir` de un elemento en línea. Siguiendo con el ejemplo anterior:

```
inglés1 HEBREO2 inglés3 HEBREO4 inglés5 HEBREO6
```

Si el idioma predominante del documento es el inglés, y el párrafo contiene una frase en hebreo que va desde HEBREO2 hasta HEBREO4, que a su vez contiene una cita en inglés (inglés3). La presentación de este texto sería:

```
inglés1 4OERBEH inglés3 2OERBEH inglés5 6OERBEH
          ----->
              I
          <-----
              H
          ----->
              I
```

Para lograr dos cambios de dirección por inclusión hay que delimitar la segunda inclusión explícitamente, utilizando el elemento `SPAN` y el atributo `dir` para dar formato al texto:

```
inglés1 <span dir="RTL">HEBREO2 inglés3 HEBREO4</span> inglés5 HEBREO6
```

EFFECTO DE LAS HOJAS DE ESTILO EN LA BIDIRECCIONALIDAD

El atributo `DIR` se basa en la distinción entre elemento en línea y en bloque, por eso hay que tener cuidado al cambiar la representación visual de un elemento a través de las hojas de estilo.

Cuando un elemento en línea que no tiene un atributo `DIR` se transforma en un elemento en bloque, hereda el atributo `DIR` del elemento padre. Y cuando un elemento en bloque que no tiene un atributo `DIR` se transforma en un elemento en línea, la presentación resultante debería ser equivalente al formato obtenido añadiendo al elemento un atributo `DIR` (igual al valor heredado).

3. Tablas y agentes de usuario no visuales



ASOCIACIÓN DE INFORMACIÓN DE ENCABEZADO CON CELDAS DE DATOS

Los sintetizadores de voz o dispositivos Braille pueden representar las celdas de forma más intuitiva cuando a las mismas se asignan atributos **headers**, **id**, **scope** y **abbr**. Cabe aclarar que se llega al mismo resultado utilizando el atributo **scope**, o la dupla **id/headers**.

Como un resumen de lo explicado en la sección de celdas de datos y encabezado:

- **id/headers:** Se asigna un nombre a través del atributo **id** a cada celda de encabezado y luego se asigna a las celdas de datos el atributo **headers** cuyo valor será el nombre asignado a la celda de encabezado que le corresponde. El atributo **id** puede ser aplicado a celdas de datos (**TD**) cuando es difícil distinguir entre datos y encabezado. **headers** es útil cuando los encabezados no se encuentran alineados con los datos a los que aplica.
- **scope:** Se asigna a las celdas de encabezado el atributo **scope** que indicará a cuáles celdas de datos proporciona información.
- **abbr:** Especifica un encabezado abreviado para celdas de encabezado de modo que los agentes de usuario puedan representar la información más rápidamente.

Ejemplos

Id/headers

CÓDIGO

Tomando una de las tablas utilizadas con anterioridad, asignamos información de encabezado a celdas por medio del atributo **headers**. Todas las celdas de una misma columna se refieren a la misma celda de

encabezado (a través del atributo **id**).

```
<table border="1" summary="Esta tabla muestra el número y tipo de bolitas ganadas
por los niños del barrio">
<caption>Bolitas ganadas</caption>
<tr>
  <th id="encabezado1">Nombre</th>
  <th id="encabezado2" abbr="Cantidad">Cantidad de bolitas</th>
  <th id="encabezado3" abbr="Tipo">Tipo de bolitas</th>
</tr>
<tr>
  <td headers="encabezado1">Sebastián</td>
  <td headers="encabezado2">10</td>
  <td headers="encabezado3">Pequeñas</td>
</tr>
<tr>
  <td headers="encabezado1">Mariano</td>
  <td headers="encabezado2">2</td>
  <td headers="encabezado3">Grandes</td>
</tr>
</table>
```

Nota: Se abrevia el encabezado "Tipo de bolitas" y "Cantidad de bolitas" a "Tipo" y "Cantidad" respectivamente, usando el atributo **abbr**.

Scope

CÓDIGO

El siguiente es el mismo ejemplo con atributo **scope**, el cuál a través de su valor "col" indica que afecta a todas "las celdas de la columna".

```
<table border="1" summary="Esta tabla muestra el número y tipo de bolitas ganadas
por los niños del barrio">
<caption>Bolitas ganadas</caption>
<tr>
  <th scope="col">Nombre</th>
  <th scope="col" abbr="Cantidad">Cantidad de bolitas</th>
  <th scope="col" abbr="Tipo">Tipo de bolitas</th>
</tr>
<tr>
  <td>Sebastián</td>
  <td>10</td>
  <td>Pequeñas</td>
</tr>
<tr>
  <td>Mariano</td>
  <td>2</td>
  <td>Grandes</td>
</tr>
</table>
```

Representación por sintetizador de voz

Los ejemplos anteriores serían representados por un sintetizador de voz, de la siguiente manera.

Título: Bolitas ganadas

Resumen: Esta tabla muestra el número y tipo de bolitas ganadas por los niños del barrio

Nombre: Sebastián, **Cantidad:** 10, **Tipo:** Pequeñas

Nombre: Mariano, **Cantidad:** 2, **Tipo:** Grandes

CATEGORIZACIÓN DE CELDAS

Consideremos la siguiente tabla:

Gastos Mensuales (por bimestre)

	Comida	Transporte	Otros	subtotales
Enero				
Semana 1	110.05	15.20	0.00	
Semana 2	25.30	22.30	15.00	
Semana 3	55.80	68.90	28.40	
Semana 4	35.00	5.15	62.00	
subtotales	226.15	111.55	105.40	443.10
Febrero				
Semana 1	324.60	75.20	41.00	
Semana 2	28.40	32.10	58.25	
Semana 3	15.00	55.32	95.12	
Semana 4	68.90	12.22	75.46	
subtotales	436.90	174.84	269.83	881.57
TOTALES	663.05	286.39	375.23	1324.67

Un usuario que navega la misma a través de un agente de usuario basado en voz, querrá obtener ciertos datos específicos de la tabla, en lugar de una descripción de la misma en base a sus encabezados. Esto es, a través de ciertas preguntas tales como:

- "¿Cuál fue el gasto en transporte?"
- "¿Cuál fue el gasto en comida la primer semana de enero?"
- "¿Cuál fue el gasto total en Enero?"

Como cada pregunta implica una relación entre varias celdas, las mismas deben estar categorizadas para que el agente de usuario sepa en cuáles de ellas buscar la información.

En este ejemplo concreto, el autor debería agrupar:

- las celdas "Enero" y "Febrero" en la categoría "Meses"
- las celdas "Comida", "Transporte" y "Otros" en la categoría "Gastos"
- las celdas "Semana 1" y demás en la categoría "Semanas"

De esta forma, el agente de usuario debería considerar las preguntas anteriormente desarrolladas como:

- "¿Cuáles celdas están en la categoría GASTOS : Transporte?"
- "¿Cuáles celdas están en la categoría (GASTOS:Comida), (MES:Enero), (SEMANA:Semana 1)?"
- "¿Cuáles celdas están en la categoría (GASTOS:Comida, Transporte, Otros), (MES:Enero)?"

La categorización se realiza estableciendo el atributo **axis** a una celda de encabezado o datos y un nombre único con el atributo **id**. Por lo tanto, la celda Comida se pondría en la categoría "Gastos" de la siguiente forma:

```
<th id="gastos1" axis="Gastos">Comida</th>
```

Así, aquellas celdas que estén relacionadas con el encabezado "Comida", deberán referirse a este último, por medio de los atributos **headers** o **scope**. De esta forma los gastos en Comida en la primer semana de enero

deberán codificarse como **headers="gastos1"** o **scope="gastos1"** porque ese es el nombre asignado a través del atributo **id** a dicho encabezado.

```
<td headers="gastos1">110.05</td>
```

La tabla representada al principio del capítulo, sería codificada de la siguiente manera si agregáramos las categorías:

```
<table border="1" summary="Gastos de Enero y Febrero en Comida, Transporte y Otros">
<caption>Gastos mensuales (por bimestre)</caption>
<tr>
<th></th>
<th axis="gastos" id="gastos1">Comida</th>
<th axis="gastos" id="gastos2">Transporte</th>
<th axis="gastos" id="gastos3">Otros</th>
<td>subtotales</td>
</tr>
<tr>
<th axis="mes" id="mes1">Enero</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
<tr>
<td axis="semana" id="semana1">Semana 1</td>
<td headers="mes1 semana1 gastos1">110.05</td>
<td headers="mes1 semana1 gastos2">15.20</td>
<td headers="mes1 semana1 gastos3">0.00</td>
<td></td>
</tr>
<tr>
<td axis="semana" id="semana2">Semana 2</td>
<td headers="mes1 semana2 gastos1">25.30</td>
<td headers="mes1 semana2 gastos2">22.30</td>
<td headers="mes1 semana2 gastos3">15.00</td>
<td></td>
</tr>
<tr>
<td axis="semana" id="semana3">Semana 3</td>
<td headers="mes1 semana3 gastos1">55.80</td>
<td headers="mes1 semana3 gastos2">68.90</td>
<td headers="mes1 semana3 gastos3">28.40</td>
<td></td>
</tr>
<tr>
<td axis="semana" id="semana4">Semana 4</td>
<td headers="mes1 semana4 gastos1">35.00</td>
<td headers="mes1 semana4 gastos2">5.15</td>
<td headers="mes1 semana4 gastos3">62.00</td>
<td></td>
</tr>
<tr>
<td>subtotales</td>
<td>226.15</td>
<td>111.55</td>
<td>105.40</td>
<td>443.10</td>
</tr>
<tr>
<th axis="mes" id="mes2">Enero</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
```

```

</tr>
<tr>
  <td axis="semana" id="semana5">Semana 1</td>
  <td headers="mes2 semana5 gastos1">324.60</td>
  <td headers="mes2 semana5 gastos2">75.20</td>
  <td headers="mes2 semana5 gastos3">41.00</td>
  <td></td>
</tr>
<tr>
  <td axis="semana" id="semana6">Semana 2</td>
  <td headers="mes2 semana6 gastos1">28.40</td>
  <td headers="mes2 semana6 gastos2">32.10</td>
  <td headers="mes2 semana6 gastos3">58.25</td>
  <td></td>
</tr>
<tr>
  <td axis="semana" id="semana7">Semana 3</td>
  <td headers="mes2 semana7 gastos1">15.00</td>
  <td headers="mes2 semana7 gastos2">55.32</td>
  <td headers="mes2 semana7 gastos3">95.12</td>
  <td></td>
</tr>
<tr>
  <td axis="semana" id="semana8">Semana 4</td>
  <td headers="mes2 semana8 gastos1">68.90</td>
  <td headers="mes2 semana8 gastos2">12.22</td>
  <td headers="mes2 semana8 gastos3">75.46</td>
  <td></td>
</tr>
<tr>
  <td>subtotales</td>
  <td>436.90</td>
  <td>174.84</td>
  <td>269.83</td>
  <td>881.57</td>
</tr>
<tr>
  <th>TOTALES</th>
  <th>663.05</th>
  <th>286.39</th>
  <th>375.23</th>
  <th>1324.67</th>
</tr>
</table>

```

🔗 [Ver ejemplo de tabla para agentes no visuales \(CD > ejemplos > xhtml > tabla > agentesnovisuales.html\)](#)

De este modo, cada atributo **headers** proporciona una lista de referencias a valores **id**. Una celda puede ser asignada a varias categorías, por lo tanto en la celda correspondiente al gasto en comidas de la primer semana de enero, **headers** debería tener los valores "**mes1 semana1 gastos1**" que corresponden con Enero, Semana 1 y Comida respectivamente.

No hay límites en cuanto a la forma en que puede categorizarse la información de una tabla, y en relación a nuestra tabla, podríamos añadir las categorías: "subtotales" y "totales".

Sin este tipo de categorización, un agente de usuario frente a la pregunta "Cuál es el gasto en comida" podría erróneamente representar todas las celdas bajo la columna Comida, entre las cuáles estarían erróneamente incluidas las de subtotales y totales.

Si tuviera que responder a la pregunta "¿Cuál fue el gasto en comida de enero?", un agente de usuario no visual debería en primera instancia buscar la información relacionada en las categorías (GASTOS:Comida), (MES:Enero), lo cuál daría como resultado (Semana 1 = 110.05, Semana 2 = 25.30, Semana 3 = 55.80, Semana 4 = 35.00). Luego, podrían representar la información de la siguiente manera:

Mes: Enero. Semana: Semana 1. Gastos, Comida: 110.05
Mes: Enero. Semana: Semana 1. Gastos, Comida: 25.30
Mes: Enero. Semana: Semana 1. Gastos, Comida: 55.80
Mes: Enero. Semana: Semana 1. Gastos, Comida: 35.00

O de forma más compacta:

Enero, Comida, Semana 1: 110.05
Semana 2: 25.30
Semana 3: 55.80
Semana 4: 35.00

4. Hoja de estilo por defecto para HTML 4.0



```
html, address,
blockquote,
body, dd, div,
dl, dt, fieldset, form,
frame, frameset,
h1, h2, h3, h4,
h5, h6, noframes,
ol, p, ul, center,
dir, hr, menu, pre    { display: block }
li                    { display: list-item }
head                  { display: none }
table                 { display: table }
tr                    { display: table-row }
thead                 { display: table-header-group }
tbody                 { display: table-row-group }
tfoot                 { display: table-footer-group }
col                   { display: table-column }
colgroup              { display: table-column-group }
td, th                { display: table-cell }
caption               { display: table-caption }
th                    { font-weight: bolder; text-align: center }
caption               { text-align: center }
body                  { margin: 8px }
h1                    { font-size: 2em; margin: .67em 0 }
h2                    { font-size: 1.5em; margin: .75em 0 }
h3                    { font-size: 1.17em; margin: .83em 0 }
h4, p,
blockquote, ul,
fieldset, form,
ol, dl, dir,
menu                  { margin: 1.12em 0 }
h5                    { font-size: .83em; margin: 1.5em 0 }
h6                    { font-size: .75em; margin: 1.67em 0 }
h1, h2, h3, h4,
h5, h6, b,
```

```

strong          { font-weight: bolder }
blockquote      { margin-left: 40px; margin-right: 40px }
i, cite, em,
var, address    { font-style: italic }
pre, tt, code,
kbd, samp       { font-family: monospace }
pre             { white-space: pre }
button, textarea,
input, select   { display: inline-block }
big             { font-size: 1.17em }
small, sub, sup { font-size: .83em }
sub             { vertical-align: sub }
sup            { vertical-align: super }
table           { border-spacing: 2px; }
thead, tbody,
tfoot           { vertical-align: middle }
td, th         { vertical-align: inherit }
s, strike, del { text-decoration: line-through }
hr             { border: 1px inset }
ol, ul, dir,
menu, dd        { margin-left: 40px }
ol             { list-style-type: decimal }
ol ul, ul ol,
ul ul, ol ol    { margin-top: 0; margin-bottom: 0 }
u, ins         { text-decoration: underline }
br:before      { content: "\A" }
:before, :after { white-space: pre-line }
center         { text-align: center }
:link, :visited { text-decoration: underline }
:focus         { outline: thin dotted invert }

/* Begin bidirectionality settings (do not change) */
BDO[DIR="ltr"] { direction: ltr; unicode-bidi: bidi-override }
BDO[DIR="rtl"] { direction: rtl; unicode-bidi: bidi-override }

*[DIR="ltr"]   { direction: ltr; unicode-bidi: embed }
*[DIR="rtl"]   { direction: rtl; unicode-bidi: embed }

@media print {
  h1           { page-break-before: always }
  h1, h2, h3,
  h4, h5, h6   { page-break-after: avoid }
  ul, ol, dl   { page-break-before: avoid }
}

```

5. Tabla de elementos



Elemento	Atributos		Modelo de contenido	
	Aceptados en XHTML strict	Desaprobados (se usan en marcos y transicional)	Acepta en Strict	No acepta
	atributo* -> el atributo es REQUERIDO		expr ? = 0 o 1 instancias permitidas	
	atributo (tipo) -> el parentesis indica el tipo de valor.		expr + = 1 o + instancias permitidas	
	atributo ("xx" "xx") -> valores legales entre comillas y separados por		expr * = 0 o + instancias permitidas	
	atributo ("xx"* "xx") -> valor legal por defecto con asterisco detrás		a , b = a es requerido seguido de b	
	atributo (URI ="xx") -> valor legal fijo luego del signo igual y entre comillas		a b = a o b son requeridos	
			a - b = a es permitida omitiendo b	
ENCABEZADO DEL DOCUMENTO				
html	i18n, id (ID), xmlns (URI ="http://www.w3.org/1999/xhtml")		head, body	
head	i18n, id (ID), profile (URIs)		title base (script style meta link object)*	
title	i18n, id (ID)		PCDATA	

Elemento	Atributos		Modelo de contenido	
	Aceptados en XHTML strict	Desaprobados (se usan en marcos y transicional)	Acepta en Strict	No acepta
base	href* (URI), id (ID)	target (FrameTarget)	EMPTY	
meta	i18n, id (ID), http-equiv (CDATA), name (CDATA), content* (CDATA), scheme (CDATA)		EMPTY	
link	attrs, charset (Charset), href (URI), hreflang (LanguageCode), type (ContentType), rel (LinkTypes), rev (LinkTypes), media (MediaDesc)	target (FrameTarget)	EMPTY	
style	i18n, id (ID), type* (ContentType), media (MediaDesc), title (Text), xml:space (= "preserve")		PCDATA	
script	id (ID), charset (Charset), type* (ContentType), src (URI), defer ("defer"), xml:space (= "preserve")	language (CDATA)	PCDATA	
noscript	attrs		Block	
CUERPO DEL DOCUMENTO				
body	attrs, onload (Script), onunload (Script)	alink (Color), background (URI), bgcolor (Color), link (Color), text (Color), vlink (Color)	Block	
div	attrs	align (TextAlign)	Flow	
p	attrs	align (TextAlign)	Inline	
h1	attrs	align (TextAlign)	Inline	
h2	attrs	align (TextAlign)	Inline	
h3	attrs	align (TextAlign)	Inline	
h4	attrs	align (TextAlign)	Inline	
h5	attrs	align (TextAlign)	Inline	
h6	attrs	align (TextAlign)	Inline	
ul	attrs	compact ("compact"), type ("disc" "square" "circle")	(li)+	
ol	attrs	compact ("compact"), start (Number), type ("arabic numbers" "lower alpha" "upper alpha" "lower roman" "upper roman")	(li)+	
li	attrs	type (CDATA), value (Number)	Flow	
dl	attrs	compact ("compact")	(dt dd)+	
dt	attrs		Inline	

Elemento	Atributos		Modelo de contenido	
	Aceptados en XHTML strict	Desaprobados (se usan en marcos y transicional)	Acepta en Strict	No acepta
dd	attrs		Flow	
address	attrs		Inline	
hr	attrs	align ("left" "center" "right"), noshade ("noshade"), size (Pixels), width (Length),	EMPTY	
pre	attrs, xml:space (= "preserve")	width (Number)	Inline	img object big small sub sup
blockquote	attrs, cite (URI)		Block	
ins	attrs, cite (URI), datetime (Datetime)		Flow	
del	attrs, cite (URI), datetime (Datetime)		Flow	
a	attrs, focus, charset (Charset), type (ContentType), name (NMToken), href (URI), coords (Coords), hreflang (LanguageCode), rel (LinkTypes), rev (LinkTypes), shape ("rect" "circle" "poly" "default")	target (FrameTarget)	Inline	a
ELEMENTOS EN LÍNEA				
span	attrs		Inline	
bdo	coreattrs, events, lang (LanguageCode), xml:lang (LanguageCode), dir* (ltr rtl)		Inline	
br	coreattrs	clear ("left" "all" "right" "none"*)	EMPTY	
em	attrs		Inline	
strong	attrs		Inline	
dfn	attrs		Inline	
code	attrs		Inline	
samp	attrs		Inline	
kbd	attrs		Inline	
var	attrs		Inline	
cite	attrs		Inline	
abbr	attrs		Inline	
acronym	attrs		Inline	
q	attrs, cite (URI)		Inline	
sub	attrs		Inline	
sup	attrs		Inline	
tt	attrs		Inline	
i	attrs		Inline	
b	attrs		Inline	
big	attrs		Inline	
small	attrs		Inline	
OBJETOS E IMÁGENES				

Elemento	Atributos		Modelo de contenido	
	Aceptados en XHTML strict	Desaprobados (se usan en marcos y transicional)	Acepta en Strict	No acepta
object	attrs, declare ("declare"), classid (URI), codebase (URI), data (URI), type (ContentType), codetype (ContentType), archive (UriList), standby (Text), height (Length), width (Length), usemap (URI), name (NMToken), tabindex (Number)	align (ImgAlign), border (Pixels), hspace (Pixels), vspace (Pixels)	(param Flow)*	
param	id (ID), name (CDATA), value (CDATA), valuetype ("data" "ref" "object"), type (ContentType)		EMPTY	
img	attrs, src* (URI), alt* (Text), longdesc (URI), height (Length), width (Length), usemap (URI), ismap (ismap)	align (ImgAlign), border (Pixels), hspace (Pixels), vspace (Pixels), name (NMToken)	EMPTY	
map	attrs, name (NMToken)		(Block+ area+)	
area	attrs, focus, shape ("rect" "circle" "poly" "default"), coords (Coords), href (URI), nohref ("nohref") alt* (Text)	target (FrameTarget)	EMPTY	
FORMULARIOS				
form	attrs, action* (URI), method ("get" "post"), enctype (ContentType="application/x-www-form-urlencoded"), onsubmit (Script), onreset (Script), accept (ContentTypes), accept-charset (Charsets)	name (NMToken), target (FrameTarget)	Block	form
label	attrs, for (IDREF), accesskey (Character), onfocus (Script), onblur (Script)		Inline	label
input	attrs, focus, name (CDATA), value (CDATA), checked ("checked"), disabled ("disabled"), size (CDATA), type ("text" "password" "checkbox" "radio" "submit" "reset" "file" "hidden" "image" "button"), readonly ("readonly"), maxlength (Number), src (URI), alt (CDATA), usemap (URI), onselect (Script), onchange (Script), accept (ContentTypes)	align (ImgAlign)	EMPTY	
select	attrs, name (CDATA), size (Number), multiple ("multiple"), disabled ("disabled"), tabindex (Number), onfocus (Script), onblur (Script), onchange (Script)		(optgroup option)+	
optgroup	attrs	disabled ("disabled"), label (Text)	(option)+	
option	attrs, selected ("selected"), disabled ("disabled"), label (Text), value (CDATA)		PCDATA	
textarea	attrs, focus, name (CDATA), rows (Number), cols (Number), disabled ("disabled"), readonly ("readonly"), onselect (Script), onchange (Script)		PCDATA	
fieldset	attrs		(legend Flow)*	

Elemento	Atributos		Modelo de contenido	
	Aceptados en XHTML strict	Desaprobados (se usan en marcos y transicional)	Acepta en Strict	No acepta
legend	attrs, accesskey (Character)	align ("left" "right" "top" "bottom")	Inline	
button	attrs, focus, name (CDATA), value (CDATA), type ("button" "submit"* "reset"), disabled ("disabled")		Flow	a form input select textarea label button fieldset iframe
TABLAS				
table	attrs, summary (Text), width (Length), border (Pixels), cellpadding (Length), cellspacing (Length), frame ("void" "above" "below" "hsides" "lhs" "rhs" "vsides" "box" "border"), rules ("none" "groups" "rows" "cols" "all")	align ("left" "center" "right"), bgcolor (Color)	(caption?, (col* colgroup*), thead?, tfoot?, (tbody+ tr+))>	
caption	attrs	align ("top" "bottom" "left" "right")	Inline	
thead	attrs, valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)		(tr)+	
tfoot	attrs, valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)		(tr)+	
tbody	attrs, valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)		(tr)+	
colgroup	attrs, span (Number), width (MultiLength), valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)		(col)*	
col	attrs, span (Number), width (MultiLength), valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)		EMPTY	
tr	attrs, valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)	bgcolor (Color)	(th td)+	
th	attrs, abbr (Text), axis (CDATA), headers (IDREFS), scope ("row" "col" "rowgroup" "colgroup"), rowspan (Number), colspan (Number), valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)	bgcolor (Color), height (Length), nowrap ("nowrap"), width (Length)	Flow	

Elemento	Atributos		Modelo de contenido	
	Aceptados en XHTML strict	Desaprobados (se usan en marcos y transicional)	Acepta en Strict	No acepta
td	attrs, abbr (Text), axis (CDATA), headers (IDREFS), scope ("row" "col" "rowgroup" "colgroup"), rowspan (Number), colspan (Number), valign ("top" "middle" "bottom" "baseline"), align ("left" "center" "right" "justify" "char"), charoff (Length), char (Character)	bgcolor (Color), height (Length), nowrap ("nowrap"), width (Length)	Flow	
MARCOS				
frameset	coreattrs, cols (MultiLength), rows (MultiLength), onload (Script), onunload (Script)		(frameset frame noframes)*	
frame	coreattrs, frameborder ("1" "0"), longdesc (URI), marginheight (Pixels), marginwidth (Pixels), noresize ("noresize"), scrolling ("yes" "no" "auto" "*"), src (URI)		EMPTY	
noframes	attrs		Flow	
TRANSICIONAL				
iframe		coreattrs, longdesc (URI), src (URI), frameborder ("1" "0"), marginwidth (Pixels), marginheight (Pixels), scrolling ("yes" "no" "auto" "*"), height (Length), width (Length), align (ImgAlign), name (NMTOKEN)	Flow	
noframes	attrs		Flow	
DESAPROBADOS (APARECEN EN FRAMESET Y TRANSICIONAL)				
basefont		color (Color), face (CDATA), id (ID), size (CDATA)	EMPTY	
center		attrs	Flow	
dir		attrs, compact ("compact")	(li)+	
font		coreattrs, l18N, color (Color), face (CDATA), size (CDATA)	Inline	
isindex		coreattrs, l18N, prompt (Text)	EMPTY	
menu		attrs, compact ("compact")	(li)+	
s		attrs	Inline	

Elemento	Atributos		Modelo de contenido	
	Aceptados en XHTML strict	Desaprobados (se usan en marcos y transicional)	Acepta en Strict	No acepta
strike		attrs	Inline	
u		attrs	Inline	
applet		coreattrs, codebase (URI), archive (UriList), code (CDATA), object (CDATA), alt (Text), name (NMToken), height (Length), width (Length), align (ImgAlign), hspace (Pixels), vspace (Pixels)	(param Flow)*	

6. Tabla de atributos



En la columna DTD, la letra L representa al DTD Transicional y la letra F al DTD para marcos. Todos los atributos para el DTD transicional se encuentran desaprobadados, salvo el atributo target y los atributos width y height para el elemento IFRAME.

Si el valor por defecto del atributo es:

- **implícito** (palabra clave #IMPLIED), el valor por defecto debe ser proporcionado por el agente de usuario (en algunos casos heredándolo de elementos padre)
- **siempre requerido** (palabra clave #REQUIRED)
- **fijo** e igual a un valor dado (palabra clave #FIXED), es explícitamente un valor por defecto para el atributo.

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
abbr	TD, TH	%Text;	#IMPLIED		abreviatura de celda de cabecera
accept-charset	FORM	%Charsets;	#IMPLIED		lista de codificaciones de caracteres soportadas
accept	FORM, INPUT	%ContentTypes;	#IMPLIED		lista de tipos MIME para subir ficheros
accesskey	A, AREA, BUTTON, INPUT, LABEL, LEGEND, TEXTAREA	%Character;	#IMPLIED		carácter de la tecla de accesibilidad
action	FORM	%URI;	#REQUIRED		procesador de formulario en servidor

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
align	CAPTION	%Align;	#IMPLIED	L	título de una tabla
	APPLET, IFRAME, IMG, INPUT, OBJECT	%IAlign;	#IMPLIED	L	alineación vertical u horizontal
	LEGEND	%LAlign;	#IMPLIED	L	leyenda de un grupo de campos (fieldset)
	TABLE	%TAlign;	#IMPLIED	L	posición de la tabla respecto a la ventana
	HR	(left center right)	#IMPLIED	L	
	DIV, H1, H2, H3, H4, H5, H6, P	(left center right justify)	#IMPLIED	L	alineación de texto
	COL, COLGROUP, TBODY, TD, TFOOT, TH, THEAD, TR	(left center right justify char)	#IMPLIED		
alink	BODY	%Color;	#IMPLIED	L	color de vínculos seleccionados
alt	APPLET	%Text;	#IMPLIED	L	descripción corta
	AREA, IMG	%Text;	#REQUIRED		descripción corta
	INPUT	CDATA	#IMPLIED		descripción corta
archive	OBJECT	CDATA	#IMPLIED		lista de URIs separados por espacios
axis	TD, TH	CDATA	#IMPLIED		lista de cabeceras relacionadas separadas por comas
background	BODY	%URI;	#IMPLIED	L	fichero de textura de fondo del documento
bgcolor	TABLE	%Color;	#IMPLIED	L	color de fondo de las celdas
	TR	%Color;	#IMPLIED	L	color de fondo de una fila
	TD, TH	%Color;	#IMPLIED	L	color de fondo de una celda
	BODY	%Color;	#IMPLIED	L	color de fondo del documento
border	TABLE	%Pixels;	#IMPLIED		controla la anchura del marco que rodea una tabla
	IMG, OBJECT	%Pixels;	#IMPLIED	L	anchura del borde de un vínculo
cellpadding	TABLE	%Length;	#IMPLIED		espacio dentro de celdas
cellspacing	TABLE	%Length;	#IMPLIED		espacio entre celdas
char	COL, COLGROUP, TBODY, TD, TFOOT, TH, THEAD, TR	%Character;	#IMPLIED		carácter de alineación, ej. char=':'
charoff	COL, COLGROUP, TBODY, TD, TFOOT, TH, THEAD, TR	%Length;	#IMPLIED		offset para carácter de alineación
charset	A, LINK, SCRIPT	%Charset;	#IMPLIED		codificación de caracteres del recurso vinculado
checked	INPUT	(checked)	#IMPLIED		para radiobotones y casillas de verificación
cite	BLOCKQUOTE, Q	%URI;	#IMPLIED		URI del documento o mensaje fuente
	DEL, INS	%URI;	#IMPLIED		información sobre la razón del cambio
class	Todos los elementos excepto BASE, BASEFONT, HEAD, HTML, META, PARAM, SCRIPT, STYLE, TITLE	CDATA	#IMPLIED		lista de clases separadas por espacios
classid	OBJECT	%URI;	#IMPLIED		identifica una implementación

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
clear	BR	(left all right none)	none	L	control del flujo de texto
code	APPLET	CDATA	#IMPLIED	L	fichero de clase applet
codebase	OBJECT	%URI;	#IMPLIED		URI base para classid, data, archive
	APPLET	%URI;	#IMPLIED	L	URI base opcional para applet
codetype	OBJECT	%ContentType;	#IMPLIED		tipo de contenido para code
color	BASEFONT, FONT	%Color;	#IMPLIED	L	color del texto
cols	FRAMESET	%MultiLengths;	#IMPLIED	F	lista de longitudes, por defecto: 100% (1 col)
	TEXTAREA	NUMBER	#REQUIRED		
colspan	TD, TH	NUMBER	1		número de columnas abarcado por celda
compact	DIR, DL, MENU, OL, UL	(compact)	#IMPLIED	L	espacio reducido entre objetos
content	META	CDATA	#REQUIRED		información asociada
coords	AREA	%Coords;	#IMPLIED		lista de longitudes separadas por coma
	A	%Coords;	#IMPLIED		para usar con mapas de imágenes en el cliente
data	OBJECT	%URI;	#IMPLIED		referencia a datos del objeto
datetime	DEL, INS	%Datetime;	#IMPLIED		fecha y hora del cambio
declare	OBJECT	(declare)	#IMPLIED		declara el objeto pero no lo crea
defer	SCRIPT	(defer)	#IMPLIED		El AU puede retrasar la ejecución del script
dir	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FRAME, FRAMESET, IFRAME, PARAM, SCRIPT	(ltr rtl)	#IMPLIED		dirección de texto débil/neutral
	BDO	(ltr rtl)	#REQUIRED		direccionalidad
disabled	BUTTON, INPUT, OPTGROUP, OPTION, SELECT, TEXTAREA	(disabled)	#IMPLIED		no disponible en este contexto
enctype	FORM	%ContentType;	"application/x-www-form-urlencoded"		
face	BASEFONT, FONT	CDATA	#IMPLIED	L	lista de nombres de fuentes separados por coma
for	LABEL	IDREF	#IMPLIED		empareja según valor de campo ID
frame	TABLE	%TFrame;	#IMPLIED		qué partes del marco representar
frameborder	FRAME, IFRAME	(1 0)	1	F	pinta bordes del marco
headers	TD, TH	IDREFS	#IMPLIED		lista de id's de celdas de encabezado
height	IFRAME	%Length;	#IMPLIED	L	altura del marco
	TD, TH	%Length;	#IMPLIED	L	altura de una celda
	IMG, OBJECT	%Length;	#IMPLIED		nueva altura
	APPLET	%Length;	#REQUIRED	L	altura inicial

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
href	A, AREA, LINK	%URI;	#IMPLIED		URI del recurso vinculado
	BASE	%URI;	#IMPLIED		URI que actúa como URI base
hreflang	A, LINK	%LanguageCode;	#IMPLIED		código de idioma
hspace	APPLET, IMG, OBJECT	%Pixels;	#IMPLIED	L	espacio de relleno horizontal
http-equiv	META	NAME	#IMPLIED		nombre de encabezado HTTP de respuesta
id	Todos los elementos excepto BASE, HEAD, HTML, META, SCRIPT, STYLE, TITLE	ID	#IMPLIED		id único en todo el documento
ismap	IMG, INPUT	(ismap)	#IMPLIED		usar mapa de imágenes en servidor
label	OPTION	%Text;	#IMPLIED		para usar en menú jerárquico
	OPTGROUP	%Text;	#REQUIRED		para usar en menú jerárquico
lang	Todos los elementos excepto APPLET, BASE, BASEFONT, BR, FRAME, FRAMESET, IFRAME, PARAM, SCRIPT	%LanguageCode;	#IMPLIED		código de idioma
language	SCRIPT	CDATA	#IMPLIED	L	nombre del lenguaje predefinido de scripts
link	BODY	%Color;	#IMPLIED	L	color de los vínculos
longdesc	IMG	%URI;	#IMPLIED		vínculo a descripción larga (complementa a alt)
	FRAME, IFRAME	%URI;	#IMPLIED	F	vínculo a descripción larga (complementa a title)
marginheight	FRAME, IFRAME	%Pixels;	#IMPLIED	F	altura del margen en píxeles
marginwidth	FRAME, IFRAME	%Pixels;	#IMPLIED	F	anchura del margen en píxeles
maxlength	INPUT	NUMBER	#IMPLIED		máximo de caracteres para campos de texto
media	STYLE	%MediaDesc;	#IMPLIED		diseñado para usar con estos medios
	LINK	%MediaDesc;	#IMPLIED		para representar en estos medios
method	FORM	(GET POST)	GET		método HTTP usado para enviar el formulario
multiple	SELECT	(multiple)	#IMPLIED		por defecto es selección simple

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
name	BUTTON, TEXTAREA	CDATA	#IMPLIED		
	APPLET	CDATA	#IMPLIED	L	permite a los applets encontrarse entre sí
	SELECT	CDATA	#IMPLIED		nombre del campo
	FORM	CDATA	#IMPLIED		nombre del formulario, para los scripts
	FRAME, IFRAME	CDATA	#IMPLIED	F	nombre del marco, para designarlo como destino
	IMG	CDATA	#IMPLIED		nombre de la imagen, para los scripts
	A	CDATA	#IMPLIED		vínculo destino con nombre
	INPUT, OBJECT	CDATA	#IMPLIED		enviar como parte del formulario
	MAP	CDATA	#REQUIRED		para su referencia por usemap
	PARAM	CDATA	#REQUIRED		nombre de propiedad
	META	NAME	#IMPLIED		nombre de metainformación
nohref	AREA	(nohref)	#IMPLIED		esta región no tiene acción
noresize	FRAME	(noresize)	#IMPLIED	F	permite a los usuarios redimensionar marcos
noshade	HR	(noshade)	#IMPLIED	L	
nowrap	TD, TH	(nowrap)	#IMPLIED	L	suprimir ajuste automático de líneas
object	APPLET	CDATA	#IMPLIED	L	fichero applet serializado
onblur	A, AREA, BUTTON, INPUT, LABEL, SELECT, TEXTAREA	%Script;	#IMPLIED		el elemento perdió el foco
onchange	INPUT, SELECT, TEXTAREA	%Script;	#IMPLIED		el valor del elemento fue modificado
onclick	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		se hizo clic con un botón del apuntador
ondblclick	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		se hizo doble clic con un botón del apuntador
onfocus	A, AREA, BUTTON, INPUT, LABEL, SELECT, TEXTAREA	%Script;	#IMPLIED		el foco se dirigió hacia el elemento
onkeydown	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		se pulsó una tecla

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
onkeypress	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		una tecla fue pulsada y soltada
onkeyup	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		una tecla fue soltada
onload	FRAMESET	%Script;	#IMPLIED	F	todos los marcos fueron cargados
onload	BODY	%Script;	#IMPLIED		el documento fue cargado
onmousedown	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		se pulsó un botón del apuntador
onmousemove	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		un apuntador se movió al interior del elemento
onmouseout	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		un apuntador se quitó de encima del elemento
onmouseover	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		un apuntador se movió encima del elemento
onmouseup	Todos los elementos excepto APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED		se soltó un botón de un apuntador
onreset	FORM	%Script;	#IMPLIED		el formulario fue reinicializado
onselect	INPUT, TEXTAREA	%Script;	#IMPLIED		se seleccionó parte de un texto
onsubmit	FORM	%Script;	#IMPLIED		el formulario fue enviado

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
onunload	FRAMESET	%Script;	#IMPLIED	F	se quitaron todos los marcos
	BODY	%Script;	#IMPLIED		el documento ha sido quitado
profile	HEAD	%URI;	#IMPLIED		diccionario con nombres de metainformación
prompt	ISINDEX	%Text;	#IMPLIED	L	mensaje indicador
readonly	TEXTAREA	(readonly)	#IMPLIED		
	INPUT	(readonly)	#IMPLIED		para texto y contraseñas
rel	A, LINK	%LinkTypes;	#IMPLIED		tipos de vínculos directos
rev	A, LINK	%LinkTypes;	#IMPLIED		tipos de vínculos inversos
rows	FRAMESET	%MultiLengths;	#IMPLIED	F	lista de longitudes, por defecto: 100% (1 fila)
	TEXTAREA	NUMBER	#REQUIRED		
rowspan	TD, TH	NUMBER	1		número de filas abarcado por la celda
rules	TABLE	%TRules;	#IMPLIED		líneas de división entre filas y columnas
scheme	META	CDATA	#IMPLIED		seleccionar forma de contenido
scope	TD, TH	%Scope;	#IMPLIED		campo de aplicación de una celda de cabecera
scrolling	FRAME, IFRAME	(yes no auto)	auto	F	barra de desplazamiento o no
selected	OPTION	(selected)	#IMPLIED		
shape	AREA	%Shape;	rect		controla la interpretación de las coordenadas
	A	%Shape;	rect		para usar con mapas de imágenes en el cliente
size	HR	%Pixels;	#IMPLIED	L	
	FONT	CDATA	#IMPLIED	L	[+ -]nn p.ej. size="+1", size="4"
	INPUT	CDATA	#IMPLIED		específico de cada tipo de campo
	BASEFONT	CDATA	#REQUIRED	L	tamaño de fuente base para elementos FONT
	SELECT	NUMBER	#IMPLIED		filas visibles
span	COL	NUMBER	1		los atributos de COL afectan a N columnas
	COLGROUP	NUMBER	1		número de columnas por defecto en el grupo
src	SCRIPT	%URI;	#IMPLIED		URI del script externo
	INPUT	%URI;	#IMPLIED		para campos con imágenes
	FRAME, IFRAME	%URI;	#IMPLIED	F	fuelle del contenido del marco
	IMG	%URI;	#REQUIRED		URI de la imagen a incluir
standby	OBJECT	%Text;	#IMPLIED		mensaje a mostrar mientras se carga
start	OL	NUMBER	#IMPLIED	L	número inicial de la secuencia
style	Todos los elementos excepto BASE, BASEFONT, HEAD, HTML, META, PARAM, SCRIPT, STYLE, TITLE	%StyleSheet;	#IMPLIED		información de estilo asociada

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
summary	TABLE	%Text;	#IMPLIED		propósito/estructura para salida por voz
tabindex	A, AREA, BUTTON, INPUT, OBJECT, SELECT, TEXTAREA	NUMBER	#IMPLIED		posición en el orden de tabulación
target	A, AREA, BASE, FORM, LINK	%FrameTarget;	#IMPLIED	L	representar en este marco
text	BODY	%Color;	#IMPLIED	L	color del texto del documento
title	Todos los elementos excepto BASE, BASEFONT, HEAD, HTML, META, PARAM, SCRIPT, TITLE	%Text;	#IMPLIED		título consultivo
type	A, LINK	%ContentType;	#IMPLIED		tipo de contenido consultivo
	OBJECT	%ContentType;	#IMPLIED		tipo de contenido para los datos
	PARAM	%ContentType;	#IMPLIED		tipo de contenido para el valor cuando valuetype=ref
	SCRIPT	%ContentType;	#REQUIRED		tipo de contenido para lenguaje de scripts
	STYLE	%ContentType;	#REQUIRED		tipo de contenido para lenguaje de estilos
	INPUT	%InputType;	TEXT		qué tipo de control hace falta
	LI	%LIStyle;	#IMPLIED	L	estilo de objeto de lista
	OL	%OLStyle;	#IMPLIED	L	estilo de numeración
	UL	%ULStyle;	#IMPLIED	L	estilo de gráfico de lista
	BUTTON	(button submit reset)	submit		para usar como botón de formulario
usemap	IMG, INPUT, OBJECT	%URI;	#IMPLIED		usar mapa de imágenes en el cliente
valign	COL, COLGROUP, TBODY, TD, TFOOT, TH, THEAD, TR	(top middle bottom baseline)	#IMPLIED		alineación vertical en celdas
value	INPUT	CDATA	#IMPLIED		especificar para radiobotones y casillas de verificación
	OPTION	CDATA	#IMPLIED		por defecto el contenido del elemento
	PARAM	CDATA	#IMPLIED		valor de propiedad
	BUTTON	CDATA	#IMPLIED		se manda al servidor cuando se envía
	LI	NUMBER	#IMPLIED	L	reinicializar número de secuencia
valuetype	PARAM	(DATA REF OBJECT)	DATA		cómo interpretar el valor
version	HTML	CDATA	%HTML. Version;	L	constante
vlink	BODY	%Color;	#IMPLIED	L	color de los vínculos visitados
vspace	APPLET, IMG, OBJECT	%Pixels;	#IMPLIED	L	espacio vertical

Nombre	Elementos Relacionados	Tipo	Valor por Defecto	DTD	Comentario
width	HR	%Length;	#IMPLIED	L	
	IFRAME	%Length;	#IMPLIED	L	anchura del marco
	IMG, OBJECT	%Length;	#IMPLIED		nueva anchura
	TABLE	%Length;	#IMPLIED		anchura de la tabla
	TD, TH	%Length;	#IMPLIED	L	anchura de la celda
	APPLET	%Length;	#REQUIRED	L	anchura inicial
	COL	%MultiLength;	#IMPLIED		especificación de la anchura de la columna
	COLGROUP	%MultiLength;	#IMPLIED		anchura por defecto de los COLs contenidos
	PRE	NUMBER	#IMPLIED	L	

7. Tabla de propiedades CSS



Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda	Porcentajes	Medio
'azimuth'	<angle> [[left-side far-left left center-left center center-right right far-right right-side] behind] leftwards rightwards inherit	center	Todos	Si	N/A	aural
'background-attachment'	scroll fixed inherit	scroll	Todos	No	N/A	visual
'background-color'	<color> transparent inherit	transparent	Todos	No	N/A	visual
'background-image'	<uri> none inherit	none	Todos	No	N/A	visual
'background-position'	[[<percentage> <length> left center right] [<percentage> <length> top center bottom]?] [[left center right] [top center bottom]] inherit	0% 0%	Todos	No	Referido al tamaño de la caja en sí.	visual
'background-repeat'	repeat repeat-x repeat-y no-repeat inherit	repeat	Todos	No	N/A	visual
'background'	['background-color' 'background-image' 'background-repeat' 'background-attachment' 'background-position'] inherit	Ver propiedades individuales	Todos	No	Permitido en 'background-position'	visual
'border-collapse'	collapse separate inherit	separate	Elementos 'table' e 'inline-table'	Si	N/A	visual
'border-color'	[<color> transparent]{1,4} inherit	Ver propiedades individuales		No	N/A	visual

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda	Porcentajes	Medio
'border-spacing'	<length> <length>? inherit	0	Elementos 'table' e 'inline-table'	Si	N/A	visual
'border-style'	<border-style>{1,4} inherit	Ver propiedades individuales	Todos	No	N/A	visual
'border-top' 'border-right' 'border-bottom' 'border-left'	[<border-width> <border-style> 'border-top-color'] inherit	Ver propiedades individuales	Todos	No	N/A	visual
'border-top-color' 'border-right-color' 'border-bottom-color' 'border-left-color'	<color> transparent inherit	El valor de la propiedad 'color'	Todos	No	N/A	visual
'border-top-style' 'border-right-style' 'border-bottom-style' 'border-left-style'	<border-style> inherit	none	Todos	No	N/A	visual
'border-top-width' 'border-right-width' 'border-bottom-width' 'border-left-width'	<border-width> inherit	medium	Todos	No	N/A	visual
'border-width'	<border-width>{1,4} inherit	Ver propiedades individuales	Todos	No	N/A	visual
'border'	[<border-width> <border-style> 'border-top-color'] inherit	Ver propiedades individuales	Todos	No	N/A	visual
'bottom'	<length> <percentage> auto inherit	auto	Elementos posicionados	No	Referido a la altura del bloque de contención	visual
'caption-side'	top bottom inherit	top	Elementos 'table-caption'	Si	N/A	visual
'clear'	none left right both inherit	none	Elementos a nivel de bloque	No	N/A	visual
'clip'	<shape> auto inherit	auto	Elementos posicionados absolutamente.	No	N/A	visual
'color'	<color> inherit	Depende del agente de usuario	Todos	Si	N/A	visual

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda	Porcentajes	Medio
'content'	normal none [<string> <uri> <counter> attr(<identifier>) open-quote close-quote no-open-quote no-close-quote]+ inherit	normal	Pseudo-elementos :before y :after	No	N/A	all
'counter-increment'	[<identifier> <integer>?]+ none inherit	none	Todos	No	N/A	all
'counter-reset'	[<identifier> <integer>?]+ none inherit	none	Todos	No	N/A	all
'cue-after'	<uri> none inherit	none	Todos	No	N/A	aural
'cue-before'	<uri> none inherit	none	Todos	No	N/A	aural
'cue'	['cue-before' 'cue-after'] inherit	Ver propiedades individuales	Todos	No	N/A	aural
'cursor'	[[<uri> ,]* [auto crosshair default pointer move e-resize ne-resize nw-resize n-resize se-resize sw-resize s-resize w-resize text wait help progress]] inherit	auto	Todos	Si	N/A	visual, interactivo
'direction'	ltr rtl inherit	ltr	Todos los elementos, pero ver el texto	Si	N/A	visual
'display'	inline block list-item run-in inline-block table inline-table table-row-group table-header-group table-footer-group table-row table-column-group table-column table-cell table-caption none inherit	inline	Todos	No	N/A	all
'elevation'	<angle> below level above higher lower inherit	level	Todos	Si	N/A	aural
'empty-cells'	show hide inherit	show	Elementos 'table-cell'	Si	N/A	visual
'float'	left right none inherit	none	Todos, pero ver "detalle"	No	N/A	visual
'font-family'	[[<family-name> <generic-family>] [, <family-name> <generic-family>]*] inherit	Depende del agente de usuario	Todos	Si	N/A	visual
'font-size'	<absolute-size> <relative-size> <length> <percentage> inherit	medium	Todos	Si	Referido al tamaño de la fuente del elemento padre	visual
'font-style'	normal italic oblique inherit	normal	Todos	Si	N/A	visual
'font-variant'	normal small-caps inherit	normal	Todos	Si	N/A	visual
'font-weight'	normal bold bolder lighter 100 200 300 400 500 600 700 800 900 inherit	normal	Todos	Si	N/A	visual

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda	Porcentajes	Medio
'font'	[['font-style' 'font-variant' 'font-weight']? 'font-size' [/ 'line-height']? 'font-family'] caption icon menu message-box small-caption status-bar inherit	Ver propiedades individuales	Todos	Si	Ver propiedades individuales	visual
'height'	<length> <percentage> auto inherit	auto	Todos los elementos, pero no aquellos reemplazados en línea, columnas de tablas y grupos de columnas.	No	Ver el texto	visual
'left'	<length> <percentage> auto inherit	auto	Elementos posicionados	No	Se refiere al ancho del bloque de contención	visual
'letter-spacing'	normal <length> inherit	normal	Todos	Si	N/A	visual
'line-height'	normal <number> <length> <percentage> inherit	normal	Todos	Si	Se refiere al tamaño de fuente del elemento mismo.	visual
'list-style-image'	<uri> none inherit	none	Elementos con 'display: list-item'	Si	N/A	visual
'list-style-position'	inside outside inherit	outside	Elementos con 'display: list-item'	Si	N/A	visual
'list-style-type'	disc circle square decimal decimal-leading-zero lower-roman upper-roman lower-greek lower-latin upper-latin armenian georgian lower-alpha upper-alpha none inherit	disc	Elementos con 'display: list-item'	Si	N/A	visual
'list-style'	['list-style-type' 'list-style-position' 'list-style-image'] inherit	Ver propiedades individuales	Elementos con 'display: list-item'	Si	N/A	visual
'margin-right' 'margin-left'	<margin-width> inherit	0	Todos los elementos, excepto aquellos elementos de tabla con display diferente de table-caption, table e inline-table	No	Se refiere al ancho del bloque de contención	visual

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda	Porcentajes	Medio
'margin-top' 'margin-bottom'	<margin-width> inherit	0	odos los elementos, excepto aquellos elementos de tabla con display diferente de table-caption, table e inline-table	No	Se refiere al ancho del bloque de contención	visual
'margin'	<margin-width>{1,4} inherit	Ver propiedades individuales	odos los elementos, excepto aquellos elementos con display diferente de table-caption, table e inline-table	No	Se refiere al ancho del bloque de contención	visual
'max-height'	<length> <percentage> none inherit	none	Todos los elementos, pero no aquellos reemplazados en línea, columnas de tablas y grupos de columnas.	No	Ver el texto	visual
'max-width'	<length> <percentage> none inherit	none	Todos los elementos, pero no aquellos reemplazados en línea, filas de tabla, y grupos de filas	No	Se refiere al ancho del bloque de contención	visual
'min-height'	<length> <percentage> inherit	0	Todos los elementos, pero no aquellos reemplazados en línea, columnas de tablas y grupos de columnas.	No	Ver el texto	visual
'min-width'	<length> <percentage> inherit	0	Todos los elementos, pero no aquellos reemplazados en línea, filas de tabla, y grupos de filas	No	Se refiere al ancho del bloque de contención	visual
'orphans'	<integer> inherit	2	Elementos a nivel de bloque	Si	N/A	visual, paginado

ANEXOS | Tabla de propiedades CSS

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda	Porcentajes	Medio
'outline-color'	<color> invert inherit	invert	Todos	No	N/A	visual, interactivo
'outline-style'	<border-style> inherit	none	Todos	No	N/A	visual, interactivo
'outline-width'	<border-width> inherit	medium	Todos	No	N/A	visual, interactivo
'outline'	['outline-color' 'outline-style' 'outline-width'] inherit	Ver propiedades individuales	Todos	No	N/A	visual, interactivo
'overflow'	visible hidden scroll auto inherit	visible	Elementos a nivel de bloque no reemplazados, celdas de tablas y bloques en línea.	No	N/A	visual
'padding-top' 'padding-right' 'padding-bottom' 'padding-left'	<padding-width> inherit	0	Todos los elementos, excepto table-row-group, table-header-group, table-footer-group, table-row, table-column-group y table-column	No	Se refiere al ancho del bloque de contención	visual
'padding'	<padding-width>{1,4} inherit	Ver propiedades individuales	Todos los elementos, excepto table-row-group, table-header-group, table-footer-group, table-row, table-column-group y table-column	No	Se refiere al ancho del bloque de contención	visual
'page-break-after'	auto always avoid left right inherit	auto	Elementos a nivel de bloque	No	N/A	visual, paginado
'page-break-before'	auto always avoid left right inherit	auto	Elementos a nivel de bloque	No	N/A	visual, paginado
'page-break-inside'	avoid auto inherit	auto	Elementos a nivel de bloque	Si	N/A	visual, paginado
'pause-after'	<time> <percentage> inherit	0	Todos	No	Ver el texto	aural
'pause-before'	<time> <percentage> inherit	0	Todos	No	Ver el texto	aural
'pause'	[[<time> <percentage>]{1,2}] inherit	Ver propiedades individuales	Todos	No	Ver descripción de 'pause-before' y 'pause-after'	aural
'pitch-range'	<number> inherit	50	Todos	Si	N/A	aural

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda	Porcentajes	Medio
'pitch'	<frequency> x-low low medium high x-high inherit	medium	Todos	Si	N/A	aural
'play-during'	<uri> [mix repeat]? auto none inherit	auto	Todos	No	N/A	aural
'position'	static relative absolute fixed inherit	static	Todos	No	N/A	visual
'quotes'	[<string> <string>]+ none inherit	depends on user agent	Todos	Si	N/A	visual
'richness'	<number> inherit	50	Todos	Si	N/A	aural
'right'	<length> <percentage> auto inherit	auto	Elementos posicionados	No	Se refiere al ancho del bloque de contención	visual
'speak-header'	once always inherit	once	Elementos que tienen información en la cabecera de la tabla	Si	N/A	aural
'speak-numeral'	digits continuous inherit	continuous	Todos	Si	N/A	aural
'speak-punctuation'	code none inherit	none	Todos	Si	N/A	aural
'speak'	normal none spell-out inherit	normal	Todos	Si	N/A	aural
'speech-rate'	<number> x-slow slow medium fast x-fast faster slower inherit	medium	Todos	Si	N/A	aural
'stress'	<number> inherit	50	Todos	Si		aural
'table-layout'	auto fixed inherit	auto	Elementos 'table' y 'inline-table'	No	N/A	visual
'text-align'	left right center justify inherit	a nameless value that acts as 'left' if 'direction' is 'ltr', 'right' if 'direction' is 'rtl'	Elementos a nivel de bloque, celdas de tabla y bloques en línea.	Si	N/A	visual
'text-decoration'	none [underline overline line-through blink] inherit	none	Todos	No (ver texto)	N/A	visual
'text-indent'	<length> <percentage> inherit	0	Elementos a nivel de bloque, celdas de tabla y bloques en línea.	Si	Se refiere al ancho del bloque de contención	visual
'text-transform'	capitalize uppercase lowercase none inherit	none	Todos	Si	N/A	visual
'top'	<length> <percentage> auto inherit	auto	Elementos posicionados	No	Referidos a la altura del bloque de contención	visual

Nombre	Valores permitidos	Valor inicial	Se aplica a:	Se hereda	Porcentajes	Medio
'unicode-bidi'	normal embed bidi-override inherit	normal	Todos los elementos, pero ver texto	No	N/A	visual
'vertical-align'	baseline sub super top text-top middle bottom text-bottom <percentage> <length> inherit	baseline	Elementos a nivel de línea y 'table-cell'	No	Referido al alto de la línea del elemento mismo.	visual
'visibility'	visible hidden collapse inherit	visible	Todos	Si	N/A	visual
'voice-family'	[[<specific-voice> <generic-voice>],]* [<specific-voice> <generic-voice>] inherit	depends on user agent	Todos	Si	N/A	aural
'volume'	<number> <percentage> silent x-soft soft medium loud x-loud inherit	medium	Todos	Si	Se refiere al valor heredado	aural
'white-space'	normal pre nowrap pre-wrap pre-line inherit	normal	Todos	Si	N/A	visual
'widows'	<integer> inherit	2	Elementos a nivel de bloque	Si	N/A	visual, paginado
'width'	<length> <percentage> auto inherit	auto	Todos los elementos, pero no aquellos reemplazados en línea, filas de tabla, y grupos de filas	No	Se refiere al ancho del bloque de contención	visual
'word-spacing'	normal <length> inherit	normal	Todos	Si	N/A	visual
'z-index'	auto <integer> inherit	auto	Elementos posicionados	No	N/A	visual

Glosario



Agente de usuario HTML

Un agente de usuario HTML es cualquier dispositivo que interprete documentos HTML. Los agentes de usuario incluyen navegadores visuales (de texto o gráficos), navegadores no visuales (audio, Braille), robots de búsqueda, proxies, etc.

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.

Aplicación de Usuario (AU)

Una aplicación de usuario es cualquier programa que lee e interpreta un documento escrito en el lenguaje del documento (HTML/XHTML) y aplica las hojas de estilo asociadas. Una aplicación de usuario puede mostrar un documento, leerlo en voz alta, permitir que sea impreso, convertirlo a otro formato, etc.

Atributo

Es un parámetro asociado a un elemento que consiste en un nombre y un valor (textual) asociado.

Autor

Un autor es una persona o programa que escribe o genera documentos HTML u hojas de estilo asociadas. Una **herramienta de autor** es un caso especial de autor, un programa que genera documentos y las hojas de estilo asociadas.

Caché

Es un conjunto de datos duplicados de otros originales. Cuando se accede por primera vez a un dato, se hace una copia en el caché y los accesos siguientes se realizan a dicha copia, haciendo que el tiempo de acceso al dato sea menor.

Contenido

Es el contenido asociado a un elemento en el documento fuente (los elementos que no tienen contenido son llamados vacíos). El contenido de un elemento puede incluir texto o varios sub-elementos (en cuyo caso el elemento es llamado padre de esos sub-elementos).

Contenido procesado

El contenido de un elemento después del procesamiento que se efectúa sobre él conforme a las hojas de estilo relevantes que han sido aplicadas. El contenido procesado de un *elemento sustituido** proviene de fuera del documento fuente. El contenido procesado también puede ser el texto alternativo de un elemento (ej., el valor del atributo HTML "alt"), y puede incluir objetos insertados implícita o explícitamente por la hoja de estilo, como viñetas, numeradores, etc.

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.

CSS

Siglas de **Cascading Style Sheets** (Hojas de estilo en Cascada), es un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML, XML o XHTML. Dicho de forma sencilla: separa

la estructura de un documento de su presentación.

Desaprobado

Un elemento o atributo desaprobado es aquel que ha quedado anticuado por la presencia de estructuras nuevas. Los elementos desaprobados se definen en el manual de referencia en los lugares apropiados, pero claramente marcados como desaprobados. Los elementos desaprobados pueden declararse obsoletos en versiones futuras de HTML.

Desarrollo sustentable

Tal como se acuñó en el año 1987 en una reunión de la Comisión Mundial del Medio Ambiente y el Desarrollo de la ONU y se plasmó en el "Informe Brundtland", el desarrollo sustentable es "el desarrollo que satisface las necesidades de las generaciones presentes sin comprometer la capacidad de las generaciones futuras para satisfacer sus propias necesidades." Se puede decir que implica un "desarrollo económico caracterizado por el uso eficiente de la tecnología más apropiada en la producción para evitar la contaminación o degradación ecológica, y posibilitar la explotación racional de los recursos naturales renovables y no renovables" (<http://www.definicion.org/desarrollo-sustentable>).

Dimensiones intrínsecas

El ancho y la altura definidos por el propio elemento, no impuesta por el entorno. En CSS se asume que todos y sólo los elementos sustituidos poseen dimensiones intrínsecas.

Documento

Un documento es una cadena de datos que, tras ser combinado con cualquier otra cadena a la que referencia, queda estructurado de tal manera que porta información contenida en elementos que se organizan tal y como está especificado en la correspondiente DTD.

Documento fuente

Es el documento que está codificado en algún lenguaje que representa al documento como una estructura de *elementos** y al cual se le aplica una o más hojas de estilo. Cada elemento consta de un nombre que lo identifica y de ciertos *atributos opcionales** y *contenido** (posiblemente vacío).

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.

Dislexia

1. Dificultad en el aprendizaje de la lectura, la escritura o el cálculo, frecuentemente asociada con trastornos de la coordinación motora y la atención, pero no de la inteligencia.
2. Incapacidad parcial o total para comprender lo que se lee causada por una lesión cerebral. (www.rae.es)

Elemento

Son las unidades estructurales sintácticas del lenguaje del documento (**P**, **TABLE**, **BODY**). Algunas reglas de las hojas de estilo usan el nombre de estos elementos para especificar los estilos asociados a los mismos.

Elemento Hijo

Un elemento A es llamado hijo de un elemento B si, y sólo si, B es el **padre** de A.

En el siguiente código li es hijo de ul porque ul lo contiene y es su padre:

```
<ul>
  <li>...</li>
</ul>
```

Elemento Descendiente

Un elemento A se llama descendiente de un elemento B, si (1) A es un *hijo** de B, (2) A es el hijo de un cierto elemento C que sea un descendiente de B.

```
<div>
<ul>
  <li>...</li>
</ul>
</div>
```

Aquí li es un descendiente de div, porque es hijo de ul que a su vez es descendiente de div.

Elemento Antepasado

Un elemento A se llama antepasado de un elemento B, si y solo si B es un *descendiente** de A.

```
<div>
<ul>
  <li>...</li>
</ul>
</div>
```

Con el mismo código anterior, div es un antepasado de li porque li es su descendiente.

Elemento Hermano

Un elemento A es llamado hermano de un elemento B si, y sólo si, B y A comparten el mismo elemento padre. Un elemento A es **hermano precedente** si viene antes que B en la estructura del documento. Un elemento A es hermano siguiente si viene después de B en la estructura del documento.

```
<div>
<ul>
  <li>...</li>
</ul>
<p>....</p>
</div>
```

Aquí ul y p son hermanos porque ambos son hijos de div. Entre ellos, ul es el hermano precedente porque aparece antes que p en la estructura, y por lo tanto p es el hermano siguiente porque aparece después.

Elemento precedente

Un elemento A es llamado elemento precedente de un elemento B si, y sólo si, (1) A es un *antepasado** (antecesor) de B o (2) A es *hermano precedente** de B.

```
<div>
<ul>
  <li>...</li>
</ul>
<p>....</p>
</div>
```

Div sería un elemento precedente de ul por ser un antepasado del mismo. Así como también ul es un elemento precedente de p por ser su hermano precedente.

Elemento siguiente

Un elemento A es llamado elemento siguiente de un elemento B si, y sólo si, B es un elemento precedente de A.

```
<div>
<ul>
  <li>...</li>
</ul>
<p>...</p>
</div>
```

P sería un elemento siguiente de ul, porque ul es a su vez elemento precedente de p.

Elemento sustituido

Es un elemento del cual el intérprete de CSS solo conoce las dimensiones intrínsecas. En HTML, los elementos de **IMG** y **OBJECT** pueden ser elementos sustituidos, ya que por ejemplo, el contenido del elemento de **IMG** es sustituido por la imagen que el atributo "**src**" designa.

em

Unidad de medida relativa que es igual a la altura (**font-size**) de la letra del elemento en el que se usa.

Estructura del documento

Es la estructura de elementos codificados en el documento fuente, donde cada elemento tiene exactamente un padre, con la excepción del elemento raíz (**HTML**) que no tiene ninguno.

Etiqueta

Las etiquetas son los delimitadores de inicio y final del código de un elemento. Se utilizan para marcar textos y todas deben abrirse y cerrarse. Algunas de ellas son:

```
<p>en párrafos</p>
<h1>encabezados principales</h1>
<h2>encabezados secundarios</h2>
<strong>negrita</strong>
<u>subrayado</u>
```

Eufonía

Sonoridad agradable que resulta de la acertada combinación de los elementos acústicos de las palabras. (www.rae.es)

Freelance

Un trabajador freelance o autónomo (del inglés freelancer), es una persona que trabaja de forma autónoma en una profesión, es decir, ofrece su trabajo a otros o acepta encargos de ellos, normalmente cobra por trabajo entregado y sin vinculación contractual.

Google

Es un motor de búsqueda que permite buscar temas según palabras clave y devuelve como resultado páginas que ha indexado con anterioridad y que hacen referencia al término buscado. Su sitio Web es www.google.com

Hoja de estilo

Un conjunto de declaraciones que especifican la presentación de un documento y que pueden provenir del *autor**, *usuario** o *aplicación de usuario**.

Hoja de estilo válida

La validez de una hoja de estilo depende del nivel del CSS usado para la hoja de estilo y se debe escribir según la gramática aclarada en su correspondiente especificación.

Holístico

Que propugna la concepción de cada realidad como un todo distinto de la suma de las partes que lo componen. (www.rae.es)

Hosting

El hosting o alojamiento web es el servicio que provee a los usuarios de Internet un sistema para poder almacenar información, imágenes, vídeo, o cualquier contenido accesible vía Web.

HTML

Acrónimo inglés de **H**ypertext **M**arkup **L**anguage (lenguaje de marcación de hipertexto), es un lenguaje de marcas diseñado para **estructurar** textos y **presentarlos** en forma de hipertexto, que es el formato estándar de las páginas Web.

Lenguaje de marcado

Se define como un conjunto de reglas para estructurar y dar formato a un documento electrónico. Separa un texto en los elementos en los que se compone (un párrafo, un encabezado, un texto en negrita, etc.) ya que utiliza *etiquetas* * para definir el inicio y el final de dichos elementos, y especifica las operaciones tipográficas (el formato) y funciones que debe ejecutar el programa visualizador sobre ellos.

Lenguaje del documento

Es el lenguaje de codificación del documento fuente (Ej. HTML, XHTML o SVG). CSS sólo se emplea para describir la presentación de los lenguajes de documento y no cambia la semántica subyacente de los mismos.

Lienzo

Describe en todos los medios “el espacio donde la estructura es procesada”. El lienzo es infinito por cada dimensión del espacio, pero el procesamiento tiene lugar en una región limitada del mismo, establecida por la aplicación de usuario de acuerdo con el medio al que está dirigido. Por ejemplo, las aplicaciones de usuario que procesan para pantalla generalmente imponen un ancho mínimo y un ancho inicial basado en las dimensiones del acceso visual.

Obsoleto

Un elemento o atributo obsoleto es aquél para el cual no hay garantía de soporte por parte de un agente de usuario.

PDA

Del inglés **P**ersonal **D**igital **A**ssistant, (Ayudante personal digital) es una computadora de mano originalmente diseñada como agenda electrónica (calendario, lista de contactos, bloc de notas y recordatorios) con un sistema de reconocimiento de escritura.

PHP

Acrónimo de **P**HP **H**ypertext **P**reprocessor, es un lenguaje de programación usado para la creación de contenido dinámico para sitios Web, y en aplicaciones para servidores.

Píxel

Es la menor unidad en la que se descompone una imagen digital, ya sea una fotografía, un fotograma de vídeo o un gráfico.

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 (ej. de forma auditiva, visual, impresa).

Semántica

Un código semánticamente correcto es aquel que está bien estructurado y describe el contenido, para lo cuál se requiere utilizar los elementos XHTML con el fin para el cuál fueron creados y separar el contenido de la presentación. El código semántico crea una estructura lógica más simple, que carga más rápido y que es más accesible para navegadores sin hojas de estilo, navegadores de texto, lectores de pantalla, navegadores viejos y buscadores.

Tooltip

Herramienta de ayuda visual que funciona al situar o hacer click con el cursor del ratón sobre algún elemento gráfico, mostrando una ayuda adicional para informar al usuario la finalidad del elemento sobre el que se encuentra.

Transferencia

Se refiere a la cantidad de información (tráfico de datos) que envía y recibe un sitio Web, esto incluye el tráfico de visitas a la Web, envío y recepción de correo, y transferencias por FTP.

Usuario

Un usuario es una persona que interactúa con una aplicación de usuario para ver, oír o usar de algún modo un documento y sus hojas de estilo asociadas. El usuario puede proporcionar una hoja de estilo personal que codifica sus preferencias individuales.

WebTV

Dispositivo que permite navegar por Internet mediante un receptor de televisión y una línea telefónica.

WYSIWYG

Acrónimo de **What You See Is What You Get** (en inglés, "lo que ves es lo que obtienes"). Se aplica a los procesadores de texto y otros editores de texto con formato (como los editores de HTML) que permiten escribir un documento viendo directamente el resultado final, en contraposición a otros procesadores de texto, en los que se escribe sobre una vista codificada del formato del texto. En el caso de editores de HTML este concepto se aplica a los que permiten escribir la página sobre una vista preliminar similar a la de un procesador de textos, ocupándose en este caso el programa de generar el código fuente en HTML.

XHTML

Acrónimo inglés de **eXtensible HyperText Markup Language** (lenguaje extensible de marcado de hipertexto), es el lenguaje de marcado* pensado para sustituir a HTML como estándar para las páginas Web. Tiene las mismas funcionalidades que este último, pero cumple especificaciones más estrictas. Su objetivo es lograr una Web semántica, donde la información, y la forma de presentarla estén claramente separadas.

Bibliografía



REFERENCIAS BIBLIOGRÁFICAS

- Beirut**, Michael; **Helfand**, Jessica; **Heller**, Steve y **Poynor**, Rick, *Fundamentos del Diseño Gráfico*, Ediciones Infinito, Buenos Aires, 1999.
- Bonsiepe**, Gui, *Del Objeto a la Interfase, mutaciones del diseño*, Ediciones Infinito, Buenos Aires, 1999.
- Ceschin**, Belén y **de Miguel**, Laura, *[e-zines] de la revista impresa a la digital*, Córdoba, 2003, pág.19.
- Cosgaya**, Pablo, *La legibilidad*. Apunte Facultad de Arquitectura, Diseño y Urbanismo de la Universidad de Buenos Aires.
- Costa**, Joan, *Identidad corporativa*, Editorial Trillas, México, 1999.
- De Buen Unna**, Jorge, *Manual de diseño Editorial*, Editorial Santillana, México, 2000.
- Dondis**, Donis A. *La sintaxis de la imagen. Introducción al alfabeto visual*, Ed. Gustavo Gili, Barcelona, 2003
- Gran Enciclopedia Universal Espasa Calpe*, Grupo Editorial Planeta , Buenos Aires, 2005, Tomo 35.
- Hernández Sampieri**, Roberto; **Fernández Collado**, Carlos y **Baptista Lucio**, Pilar, *Metodología de la Investigación*, Mc Graw Hill, México, 2003.
- Lynch**, Patrick y **Horton**, Sarah, *Principios de diseño básicos para la creación de sitios Web*, Ediciones Gustavo Gili, Barcelona, 2002.
- Moreno Muñoz**, Antonio, *Diseño Ergonómico de aplicaciones Hipermedia. Colección Papeles de Comunicación N°31*, Editorial Paidós, España, 2000.
- Müller-Brockmann**, Josef, *Sistemas de retículas. Un manual para diseñadores gráficos*, Gustavo Gili, México, 1992.
- Nielsen**, Jacob, *Usabilidad. Diseño de sitios Web*, Prentice Hall, Madrid, 2000.
- Orihuela**, José Luis y **Santos**, María Luisa, *Introducción al Diseño Digital. Concepción y Desarrollo de Proyectos de Comunicación Interactiva*, Anaya Multimedia, Madrid, España, 1999.
- Ruder**, Emil, *Manual de diseño tipográfico*, Gustavo Gili, México, 2º edición, 1992.
- Savino**, Carlos, *El proceso de la investigación*, Editorial Humanitas, Argentina, 1986.
- Stake**, Robert E., *Investigación con estudio de casos*, Morata, Madrid, 1999.
- Taylor**, Steven John y **Bogdan**, Robert C., *Introducción a los métodos cualitativos de investigación*, Editorial Paidós, España, 1987.
- Villafañe**, Justo, *Introducción a la teoría de la Imagen*, Editorial Pirámide, Madrid, 1985.
- Zeldman**, Jeffrey; *Designing with web Standards*, New Riders Press, 2003

REFERENCIAS WEB

Cascading Style Sheets, level 2 revision 1

CSS 2.1 Specification. W3C Working Draft 06 November 2006
<http://www.w3.org/TR/CSS21/>

Eric Meyer
<http://www.meyerweb.com>

Jeffrey Zeldman
<http://www.zeldman.com>

HTML 4.01 Specification
W3C Recommendation 24 December 1999
<http://www.w3.org/TR/html4/>

Minid.net
<http://www.minid.net>

Real Academia Española
<http://www.rae.es>


W3C - Web Style Sheets Home Page
<http://www.w3.org/Style/>

Wikipedia
<http://es.wikipedia.org>

Word reference
<http://www.wordreference.com/sinonimos/accesible>

World Wide Web Consortium
<http://www.w3.org>

XHTML™ 1.0 The Extensible HyperText Markup Language (2nd Edition)
W3C Recommendation 26 January 2000, revised 1 August 2002
<http://www.w3.org/TR/xhtml1/>



... siempre habrá una forma intuitiva, empírica y hasta fuera de toda lógica para aproximarse al Diseño Web. Muchos diseñadores continuarán, durante muchos años trabajando de esa manera. Pero, para aquellos que quieran escapar de la chapucería y comenzar a entender las nuevas formas de hacer Diseño Digital en Internet, este trabajo puede ser un excelente mapa de ruta.

Prof. Arturo Moya, Córdoba, agosto 2007.