Diseño Web — Tecnologías Involucradas

Capa de Presentación CSS

info@covetel.com.ve [1]

[1] Cooperativa Venezolana de Tecnologías Libres R.S.

Las ventajas de CSS

- Mayor control de la tipografía del diseño de la página.
- Menos trabajo
- Documentos potencialmente más pequeños
- Documentos potencialmente más accesibles
- HTML de presentación ya esta en vía de desaparecer
- Tiene buen soporte

Como funciona CSS

- Se crea el documento XHTML o HTML
- Se escriben las reglas de estilo para definir el aspecto
- Se vinculan los estilos al documento.

Hojas de estilo incrustadas <style> </style>

Un método más compacto para agregar hojas de estilo, es incrustar un bloque de estilo en la parte superior del documento HTML.

Atributos no comunes

- media='all|aural|braille|handheld|print|projection|screen|tty|tv'
- title='text'
- type='tipo de contenido' (*requerido*)

```
1  <html>
2  <head>
3  <style type="text/css">
4  h1 {color:red;}
5  p {color:blue;}
6  </style>
7  </head>
8  <body>
9  <h1>Header 1</h1>
10 <p>A paragraph.</p>
11 </body>
12 </html>
```

Hojas de estilo externas

El método mas eficiente de utilizar CSS es reunir todas las reglas de estilo en un documento de texto independiente y crear vínculos a ese documento desde todas las páginas de un sitio.

Este modo permite hacer cambios de estilo homogéneos en todo el sitio editando un solo documento.

El documento de hoja de estilo es un documento de texto con al menos una regla de estilo.

El documento de hoja de estilo puede contener comentarios, como por ejemplo:

```
1  /* Esto es un comentario */
```

Utilizar un Link

El método con mejor soporte para hacer referencia a hojas de estilo es crear un vínculo al documento CSS utilizando el elemento `link` en la cabecera `head` del documento.

```
1  <head>
2      <link rel="stylesheet" type="text/css" href="theme.css" />
3  </head>
```

El atributo `rel` define la relación del documento externo con el documento actual, el atributo `href` indica la URL del documento de hoja de estilo. Los autores pueden vincular más de una hoja de estilo al documento.

Importar

Una alternativa a los vínculos es importar una hoja de estilo externa a un documento utilizando la función `@import` en el elemento `style`

```
1  <style type="text/css">
2      <!--
3      @import url(http://www.example.com/stylesheet.css);
4      -->
5  </style>
```

Conceptos fundamentales

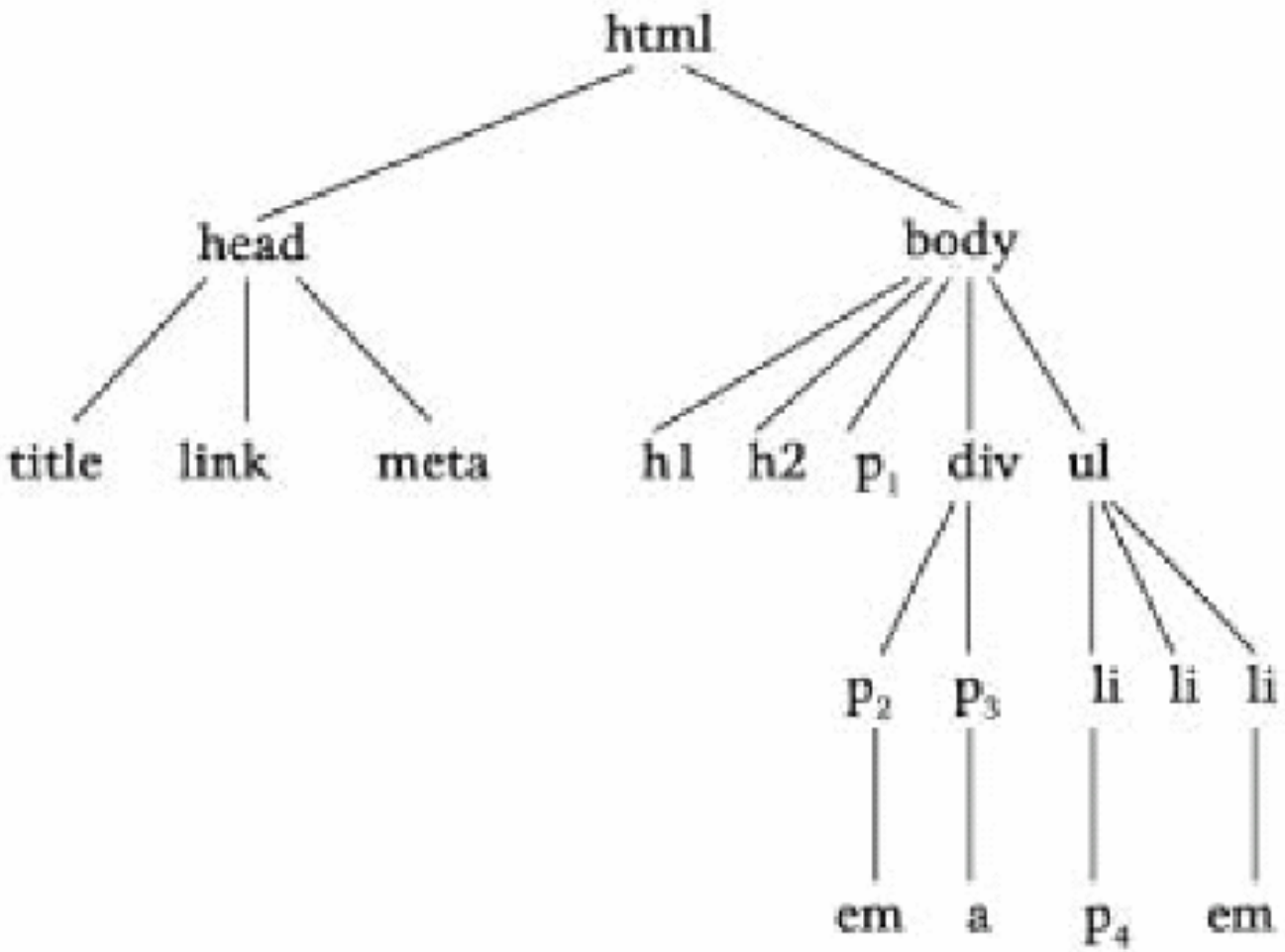
Para familiarizarse con el comportamiento de CSS es importante comprender sus conceptos claves.

- Estructura y herencia documentales
- Reglas de estilo en conflicto: la çascada"
- Tipos de elementos
- El modelo de cajas.

Estructura y herencia documentales

Los documentos XML, XHTML y HTML tienen una jerarquía implícita. Por ejemplo, el documento raíz `html` suele contener un `head` y un `body` y el `body` a su vez contiene distintos elementos a nivel de bloque como párrafos `p`. Esta jerarquía puede visualizarse como un árbol, separándose en ramas desde la raíz. La siguiente figura muestra la estructura de un documento XHTML muy sencillo.

Estructura y herencia documentales



La relación padre-hijo

La estructura del documento se convierte en un árbol genealógico cuando se hace referencia a las relaciones entre elementos. Un elemento que esta contemplado directamente por otro elemento es el **hijo** de éste. En la figura anterior el elemento `p1` es hijo de `body` y `body` es su **padre**. Los elementos que tienen el mismo padre, son **hermanos**. En el ejemplo el elemento `li` es el hijo de `ul` y el resto de los elementos `li` son sus hermanos.

La relación **padre-hijo** es fundamental para el funcionamiento de CSS

Herencia

En la relación con las relaciones estructurales está el concepto de herencia, por la cual la mayoría de los estilos pasan de un elemento a sus descendientes. Esto indica en otras palabras que un hijo puede heredar propiedades de su padre.

Por ejemplo, si una regla de estilo aplica un color para un elemento `ul`, también aplica para para todos sus hijos `li`.

En CSS la mayoría de las propiedades se heredan, pero algunas como márgenes y fondos no.

Reglas de estilo en conflicto: la cascada

Es posible y común que los elementos de un documento tengan instrucciones de presentación de distintas fuentes. Es posible entonces, que surjan conflictos. El grupo de trabajo que desarrolló CSS ya anticipó esta situación y concibió un sistema jerárquico que asigna distintos pesos a distintas fuentes de información de estilo.

La **cascada** de hojas de estilo en cascada se refiere a lo que ocurre si varias fuentes de información de estilo concurren por el control de los elementos de una página; la información de estilo se transmite hasta que es ignorada por un comando de estilo con más peso.

La cascada ordena proporcionar un conjunto de reglas para resolver conflictos entre hojas de estilo concurrentes. Cuando un agente de usuario encuentra un elemento mira a todas las declaraciones de estilo que pueda tener aplicadas y las ordena de acuerdo al origen de la hoja de estilo, a la especificidad de los selectores y al orden de la regla para determinar cuál aplicar.

Origen de la hoja de estilo

Los navegadores dan distinto peso a las hojas de estilo de estas fuentes, listadas de menor a mayor peso:

- Hojas de estilo del agente de usuario
- Hojas de estilo del lector
- Hojas de estilo del autor
- Declaraciones de estilo `!important` del lector.

Jerarquía de pesos adicionales I

Tras considerar la fuente de la hoja de estilo hay otra jerarquía de pesos que se aplica a las hojas de estilo creadas por el autor del documento.

Los puntos de origen en las hojas de estilo del autor tienen también distintos pesos (recuerde que todos los estilos del autor ignoran los estilos del agente de usuario y del lector a no ser que el lector marque el estilo como `!important`)

Esta lista señala el peso de las distintas declaraciones de estilo del autor, de menor a mayor peso. En otras palabras, las reglas de estilo que están al final de la lista ignoran a las primeras.

1. **Hojas de estilo externas vinculadas (utilizando el atributo `link`)**: Si hay varias hojas de estilo vinculadas, las reglas de estilo de las hojas de estilo listadas por debajo en el documento tendrán preferencia sobre las listadas por encima. Por ejemplo, si un documento HTML vincula a dos hojas de estilo, de este modo:

```
1  <head>
2  <link rel="stylesheet" href="style1.css" type="text/css" />
3  <link rel="stylesheet" href="style2.css" type="text/css" />
4  </head>
```

Jerarquía de pesos adicionales II

Si una regla de estilo indicada en `style2.css` esta en conflicto con una regla de estilo de `style1.css`, la regla de `style2.css` tendrá preferencia porque la hoja de estilo está listada por debajo en el documento fuente.

2. **Hojas de estilo externas importadas (utilizando `@import`)**: La información de estilo importada ignora los estilos vinculados en el header. Si hay varias directivas `@import` las reglas indicadas en las hojas de estilo que estén por debajo en la lista ignoran las que están por encima.
3. **Hojas de estilo incrustadas (con el elemento `<style>`)**: Los estilos aplicados a un documento determinado ignoran las reglas establecidas externamente.
4. **Estilos en línea (utilizando el atributo `style=` en una etiqueta de elemento)**: Los estilos en línea ignoran todas las demás declaraciones de estilo que puedan hacer referencia a ese elemento, con una excepción:

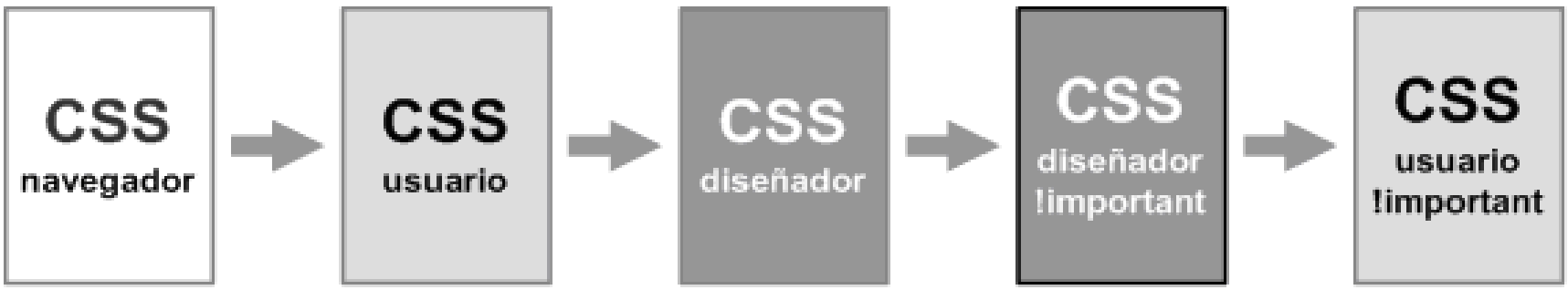
Jerarquía de pesos adicionales III

5. **Declaraciones de estilo marcadas como** `!important`: Cualquier estilo marcado como `!important` ignora todas las reglas de estilo en conflicto. Lo único que puede ignorar una regla importante de una hoja de estilo de autor es una regla importante creada por el usuario.

Ejemplo utilizando la directiva `!important`:

```
1  p {color: blue !important; }
```

En la siguiente gráfica podemos ver el orden de prioridades.



Especificidad del selector I

La cascada continua a nivel de reglas, en el siguiente ejemplo, hay dos reglas que hacen referencia al elemento `strong`

```
1  strong {color: red;}
2  h1 strong {color: blue;}
```

El agente de usuario asigna distintos tipos de niveles de peso a los distintos tipos de selectores. Cuanto más específico sea el selector más peso se le dará para ignorar las las declaraciones en conflicto. Para el ejemplo anterior todo el texto `strong` del documento se renderizará en rojo. Sin embargo, si el texto `strong` aparece dentro de una cabecera `h1` se renderizará en azul porque un elemento de un contexto determinado es más específico y tiene más peso. Esta es una lista de tipos de selector en orden de peso de menor a mayor.

1. Selectores de elementos y pseudoelementos específicos (por ejemplo `p` o `first-letter`)
2. Selectores contextuales (p.e. `h1 strong`)

Especificidad del selector II

3. Selectores de clase (p.e. `p.special`)
4. Selectores ID (p.e. `p#intro`)

Recuerde que todas las reglas marcadas como `!important` ignorarán reglas en conflicto sea cual sea su especificidad u orden.

Orden de las reglas I

Por último, cuando los estilos se han ordenado por autor, método de vinculación y especificidad, puede seguir habiendo conflictos dentro de una única fuente de hoja de estilo. Cuando una hoja de estilo contiene varias reglas en conflicto de igual peso, la que esté de última en lugar tiene más peso e ignora al resto. Por ejemplo, en el siguiente ejemplo, todas las cabeceras de primer nivel del documento serían rojas porque se impone la última regla.

```
1  h1 {color: green;}
2  h1 {color: blue;}
3  h1 {color: red;}
```

Ya se habló de este principio de **el último gana** en relación a varios elementos `link` y comandos `@import`. También se aplica en un solo bloque de declaración. Por ejemplo.

```
1  div#side {border-color: gray;
2      border-color-top: black; }
```

Elementos de bloque y en línea I

En XHTML, la distinción entre los elementos a nivel de bloque y en línea se basa en reglas de contención o, en otras palabras, en función del anidamiento de elementos en otros documentos. En general, los elementos a nivel de bloque pueden contener elementos elementos en línea y elementos de bloque, mientras que los elementos en linea solo pueden contener texto y otros elementos en linea.

Sin embargo, algunos de estos elementos de bloque deben cumplir reglas especiales en XHTML, como son los párrafos, las cabeceras, y direcciones (`address`) solo pueden contener elementos en línea y contenido.

En CSS, sin embargo, la noción de **nivel de bloque** y **en linea** es puramente de presentación visual. `block` e `inline` son dos posibles roles de maestra en pantalla utilizados para indicar a los agentes de usuario cómo presentar el documento en el diseño.

Elementos de bloque y en línea II

Un elemento a nivel de bloque de CSS (`display: block`) siempre genera saltos antes y después de él. Llena el ancho disponible del elemento padre que lo contiene sea el ancho del cuerpo del documento o un espacio menor definido como el de un `div`. No puede emplazar nada junto a un elemento de bloque en el flujo normal del documento.

Los elementos en línea CSS (`display: inline`) no generan saltos de linea. Aparecen en el flujo de la linea y sólo pasarán a otra línea si no tienen espacio.

A diferencia de las nociones de XHTML de bloque y en linea, un elemento a nivel de bloque CSS puede estar anidado en un elemento en línea y viceversa. Al utilizar CSS cualquier elemento XHTML (o XML) puede convertirse a nivel de bloque o en línea.

Introducción al modelo de cajas I

El modelo de cajas constituye la piedra angular del sistema de formateo visual de CSS. Es un concepto fundamental para comprender el funcionamiento de las hojas de estilo.

De acuerdo al modelo de cajas todos los elementos, a nivel de bloque o en línea, generan una caja rectangular alrededor llamada **caja de elemento** (aunque las cajas de bloque y de línea se tratan ligeramente de manera distinta).

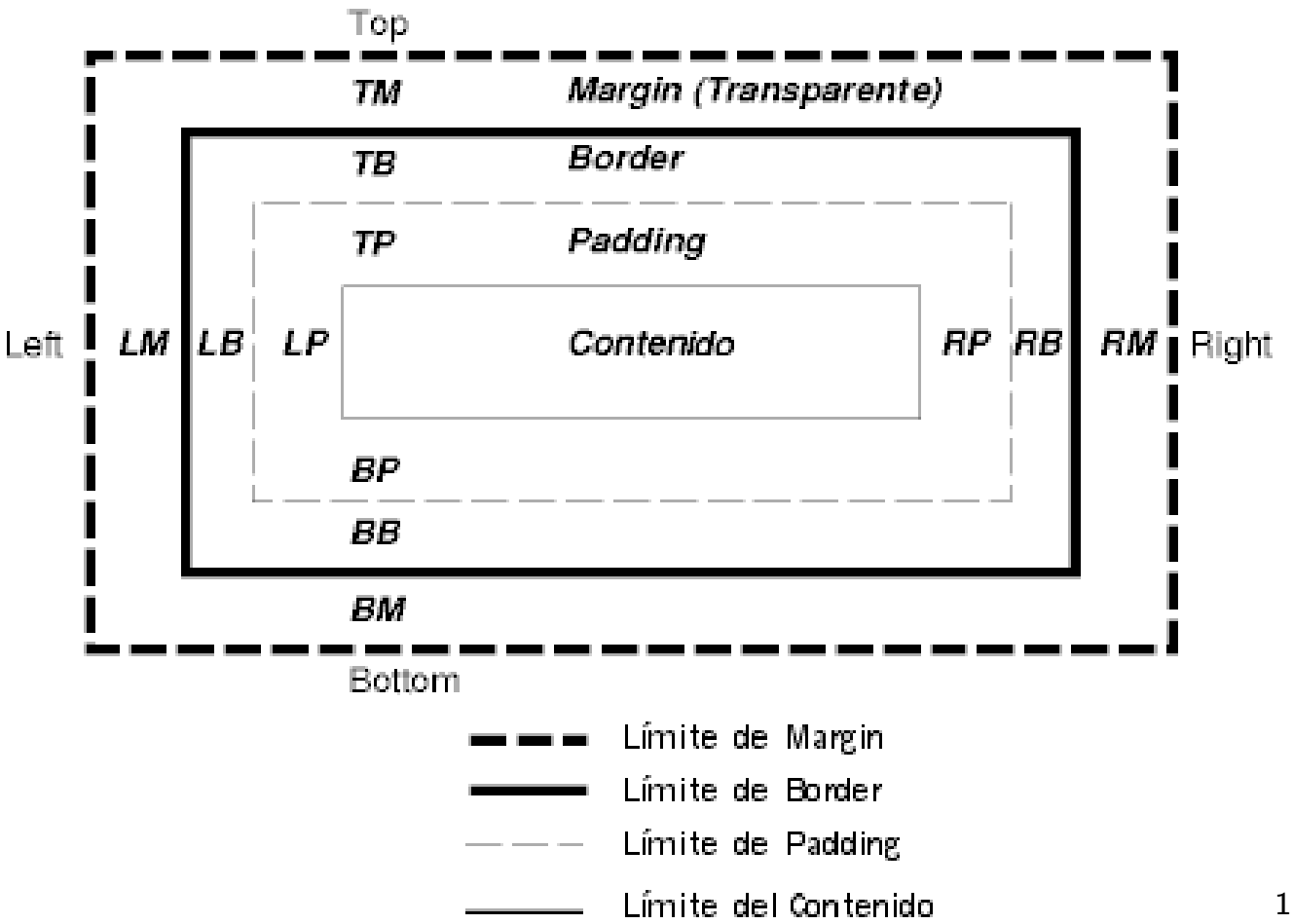
Pueden aplicarse propiedades como bordes, márgenes y fondos (entre otras) a la caja de un elemento. Las cajas también pueden utilizarse para posicionar elementos y diseñar la página.

Las cajas de elementos están hechas de cuatro componentes principales. En el núcleo de la caja está el contenido del elemento. El contenido está rodeado por cierta cantidad de relleno, sigue el borde rodeado por el margen como puede verse en las figuras 1 y 2.

Introducción al modelo de cajas II

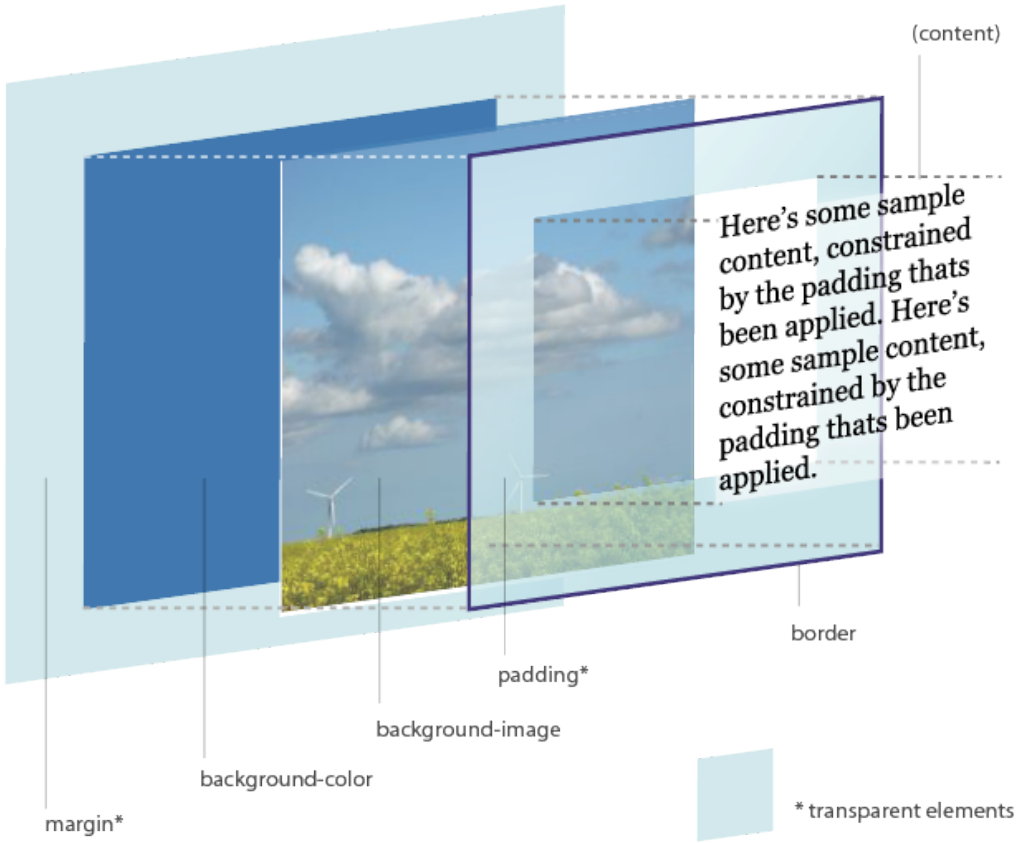
Figura: Modelo de Cajas

Introducción al modelo de cajas III



Introducción al modelo de cajas IV

Figura: Modelo de Cajas 3D



Introducción al modelo de cajas V

Característias fundamentales del modelo de caja que merece la pena destacar:

1. Relleno, bordes y márgenes son opcionales. Si ajusta a cero sus valores se eliminan de la caja.
2. El área de relleno es el espacio entre el borde del área de contenido y el borde (si lo hay). Cualquier color o imagen de fondo aplicados al elemento se extenderá por toda el área de relleno.
3. Los bordes se generan con propiedades de estilo, que especifican su estilo, ancho y color. Cuando un borde tiene huecos, el color o imagen de fondo aparece a través de esos huecos. En otras palabras, los fondos se extienden tras el borde hasta el borde exterior.
4. Los margenes siempre son transparentes, lo que significa que el color de fondo o el patrón del elemento padre se mostrará a través de ellos. El límite del margen (el borde exterior del elemento) no es visible, pero es una cantidad calculada.

Introducción al modelo de cajas VI

5. El ancho de un documento solo se aplica al ancho del área de contenido. Esto significa que si especifica que un elemento debería tener una anchura de 200px, los contenidos reales se mostrarían con una anchura de 200px, y los anchos acumulativos del relleno, el borde y los márgenes, se sumarían a esa cantidad.
6. Se puede cambiar el estilo de los lados superior, derecho, inferior, e izquierdo de una caja de elemento por separado. Por ejemplo, puede añadir un borde a la parte inferior de un elemento o solo a los lados derecho e izquierdo.

[1] Esta imágen se utiliza con autorización de `http://www.sidar.org/recur/desdi/traduc/es/css/box.html`

[2] Esta imágen se utiliza con autorización de `http://www.hicksdesign.co.uk/boxmodel/`

Unidades de longitud I

CSS permite especificar medidas en muchas unidades. Algunas de ellas como `em` y `pica` están tomadas de la prensa tradicional. Al especificar longitudes es importante recordar lo siguiente:

- No añada espacio entre el número y la abreviatura de la unidad en dos letras. Debe ser `24px` y no `24 px`
- El único valor que no necesita abreviatura de unidad es cero.
- Las medidas pueden contener fracciones decimales como `14.5cm`
- Algunas propiedades, como los márgenes, aceptan valores negativos: `margin: -500px`

En la siguiente lista puede encontrar las unidades disponibles:

`px` **(Píxel)** Las unidades de píxel son relativas a la resolución del monitor

`pt` **(Punto)** Unidad tradicional de medida para los tipos. En CSS, un punto equivale a 1/72 pulgadas.

Unidades de longitud II

`pc` **(Pica)** Unidad tradicional de medida equivalente a 12 puntos o a 1/6 pulgadas.

`em` **(Em)** Unidad de medida relativa que equivale tradicionalmente al ancho de la letra M mayúscula. En CSS equivale al tamaño en puntos de la fuente (pe. un espacio em en un tipo de 24pt mide 24 puntos de ancho). Se usa para medidas verticales y horizontales

`ex` **(Ex)** Unidad de medida relativa que es la altura de la letra minúscula **x** para esa fuente (aproximadamente la mitad de un `em`)

`in` **(Pulgadas)** Medida métrica

`mm` **(Milímetros)** Medida métrica

`cm` **(Centímetros)** Medida métrica

Especificar el color I

Al igual que en HTML, hay dos formas de especificar el color: por el nombre y por el valor numérico.

Por el nombre

Puede especificar valores cromáticos por el nombre, de este modo:

```
1  h1 {color: olive;}
```

La especificación CSS 2.1 solo acepta 17 nombres de color para utilizar en hojas de estilo (CSS 1 y CSS 2 solo tenían 16 nombres, el naranja se añadió en la versión 2.1). Los nombres de color son:

aqua	green	orange	white
black	lime	purple	yellow
blue	maroon	red	
fuchsia	navy	silver	
gray			

Especificar el color II

Por el valor rgb

En las hojas de estilo, los colores RGB pueden especificarse por cualquiera de estos métodos:

```
1  {color: #0000FF;}
2  {color: ##00F;}
3  {color: rgb(0, 0, 255);}
4  {color: rgb(0%, 0%, 100%)}
```

El primer método utiliza tres valores RGB hexadecimales de dos dígitos. El segundo método utiliza una sintaxis de tres dígitos, que se convierte a la forma de seis dígitos replicando cada dígito. (por tanto ##00F) es lo mismo que ##0000FF). Los dos últimos métodos utilizan notación funcional que especifica los valores RGB como una lista separada por comas de valores regulares de 0 a 255 o porcentuales de 0 a 100 por ciento.

Selectores

El selector es la parte de la regla de estilo que especifica el elemento o los elementos al que se aplican las instrucciones de presentación. Por ejemplo, si quiere que todos los `h1` de un documento sean verdes escriba una sola regla de estilo que tenga `h1` como selector.

Tipos de Selectores

CSS ofrece varios tipos de selectores:

- Selectores de tipo (elemento)
- Selectores contextuales (descendiente, hijo y hermano adyacente)
- Selectores de clase e ID
- Selectores de atributos
- Pseudoclases
- Pseudoelementos.

Puede ver los selectores disponibles en CSS3 en la dirección `http://www.w3.org/TR/css3-selectors/`

Selector de tipo

El selector de tipo es el selector más sencillo, selecciona un elemento por su nombre.

```
1  h1 { color: blue; }
2  h2 { color: blue; }
3  p { color: blue; }
```

Los selectores de tipo pueden agruparse en listas separadas por comas de modo que pueda aplicarse en todos ellos una sola propiedad. El siguiente ejemplo tiene el mismo efecto que el ejemplo anterior.

```
1  h1, h2, p { color: blue; }
```

CSS 2 introdujo un selector de elementos universal (*) que coincide con todos los elementos. La regla de estilo `* color: gray` pone en gris todos los elementos del documento.

Selectores contextuales

Los selectores de tipo, como los del ejemplo anterior, se aplican a todos los casos en los que se encuentre el elemento en un documento. Por contra, los selectores contextuales le permiten aplicar propiedades de estilo a los elementos seleccionados, basándose en su contexto o relación con otro elemento.

Selector descendiente

Los selectores descendiente seleccionan elementos contenidos en otro elemento. Se indican en una lista separada por un espacio, comenzando con el elemento de nivel más alto.

```
1  li em {color: olive;}
```

Como los selectores de tipo simples, los selectores contextuales pueden agruparse también en listas separadas por comas.

```
1  h1 em, h2 em, h3 em {color: red;}
```

Los selectores descendientes también pueden estar anidados a varias capas de profundidad, tal y como muestra este ejemplo que selecciona sólo texto enfatizado (`em`) en anclas (`a`) en listas ordenadas `ol`.

```
1  ol a em  {font-weight: bold;}
```

Selector hijo

Un selector hijo es parecido al selector descendiente pero sólo selecciona hijos directos de un elemento dado. Los selectores hijo están separados por el símbolo mayor que (¿).

```
1  p > em {background-color: gray;}
```

Selector hermano adyacente

Este selector se utiliza para seleccionar un elemento que sigue inmediatamente a otro elemento con el mismo elemento padre. El combinador para estos selectores es un signo más.

```
1  h1 + p {padding-left: 40;}
```

Selector class

Utilice el atributo `class` para identificar distintos elementos como parte de un grupo conceptual. Los elementos de una clase pueden modificarse con una sola regla de estilo.

```
<h1 class="special"> Attention ! </h1>
<p class="special"> You look marvelous today. </p>

h1.special {color: red;}
p.special {color: red;}


.special {color: red;}
```

Al seleccionar un nombre para una clase, asegúrese de que tenga significado. Tenga en cuenta que el nombre de una clase no puede contener espacios, y los guiones bajos están desaconsejados por la falta de soporte de algunos navegadores.

Selector ID

El selector `id` se emplea de manera similar a `class` pero se utiliza para seleccionar un único elemento en lugar de un grupo.

En el diseño Web actual, los atributos `id` suelen utilizarse para identificar secciones principales que normalmente son `div` de una página. Algunos valores frecuentes para este fin son: `content`, `header`, `sidebar`, `navigation` y `footer`.

Los nombres `id` deberían elegirse en función del papel semántico del elemento, no por su presentación. Por ejemplo, para una barra lateral a la izquierda de la página que contenga noticias es mejor llamar al div `div id='sidebar-news'` que `div id='sidebar-left'`.

Selección de atributo simple

El selector de atributo más amplio selecciona elementos con un determinado atributo, sea cual sea su valor. La sintaxis es como sigue:

```
elemento[atributo]
```

Ejemplo:

```
img[title] {border: 3px solid red;}
```

Valor de atributo exacto

Selecciona los elementos en función de un atributo con un valor de atributo exacto.

```
elemento[atributo='valorexacto']
```

Ejemplo:

```
img[title="titulo1"] {border: 3px solid red;}
```

Valor de atributo parcial

En el caso de atributos que aceptan listas de valores separados por comas, este selector de atributo permite buscar uno solo de esos valores (en lugar de la cadena entera). El signo en el selector diferencia este selector de los que coinciden con un valor exacto.

```
elemento[atributo~="valor"]
```

Ejemplo

```
img[title~='grande']
```

Valor de atributo separado por guiones

Este selector está ideado para seleccionar valores separados por guiones. El selector busca coincidencias con el valor especificado o con ese valor seguido por un guión. Se utiliza regularmente para aplicar estilo a un texto en un idioma específico.
Ejemplo:

```
*[lang|="it"] {
        font-family: SimSum-18030,SimHei, serif;
        border: 1px solid black;
        font-size: 19pt;
}

<p lang="it-CH"> Questa è la storia di un mondo ideale </p>
```

Pseudoclases

Las pseudoclases funcionan como si hubiera una clase aplicada a un grupo de elementos, en la mayoría de los casos el elemento ancla a.

Pseudoclases ancla

```
1  a:link {color: red; text-decoration: none;}
2  a:visited {color: blue;}
3  a:hover {color: fuchsia; text-decoration: underline;}
4  a:active {color: maroon;}
```

Otros Pseudoclases

Tenga cuidado, todavía no tienen buen soporte en algunos casos.

`:focus`	Selecciona elementos que tienen foco.
`:first-child`	Selecciona el primer hijo de un elemento padre.
`:lang()`	Selecciona elementos para los que se ha especificado un idioma.

Pseudo Elementos

`:first-line` Este selector aplica la regla de estilo a la primera línea del elemento especificado.

`:first-letter` Vincula un estilo a la primera letra de un elemento.

`:before` y `:after` Inserta contenido generado antes y/o después de un elemento y declaran un estilo para ese contenido. En el siguiente ejemplo inserta comillas muy marcadas antes y después de un `blockquote`.

```
1  blockquote:before {
2      content: '"';
3      font-size: 24px;
4      color: purple; }
5
6  blockquote:after {
7      content: '"';
8      font-size: 24px;
9      color: purple; }
```

Centrar una página

Existen varios métodos para hacer esto.
Veamos el primero y el modo más adecuado de centrar un elemento de ancho fijo, el cual consiste en especificar la anchura del elemento que contiene todos los contenidos de la página (normalmente un `div`) y luego ajustar los márgenes izquierdo y derecho a `auto`.

```
1  div#pagina{
2      width: 500px;
3      margin-left: auto;
4      margin-right: auto;
5  }
```

Existe otra solución menos elegante, que consiste en centrar toda la página utilizando la propiedad `text-align` en el elemento body.

```
1  body { text-align: center; }
2  body * { text-align: left; }
3
4  div#pagina {
5      width: 500px;
6      margin: 0 auto;
7  }
```

Diseños en dos columnas I

El marcado y los estilos de este ejemplo producen una página con un área de cabecera, una columna principal de contenido, una barra lateral de vínculos y un pie con información de copyleft.

Este marcado proporciona los elementos necesarios para el diseño en dos columnas.

```
<div class="masterhead"> Masterhead and headline </div>
<div class="main"> <h1> Titulo del articulo </h1> </div>
<div class="sidebar">
    <h2>Lista de enlaces</h2>
    <ul>
        <li> Enlace 1 </li>
        <li> Enlace 1 </li>
        <li> Enlace 1 </li>
    </ul>
</div>
<div class="footer"> Copyleft </div>
```

Diseños en dos columnas II

El div con contenido fue situado de primero antes que el div con la lista de links para que los usuarios con navegadores no gráficos accedan primero al el. Las reglas de estilo que se ocupan del flotamiento son éstas:

```
.masterhead {
    background: #ccc;
    padding: 15px;
}
.main {
    float: left;
    width: 70%;
    margin-right: 3%;
    margin-left: 3%;
}
.footer{
    clear: left;
    padding: 15px;
    background: #666;
}
```