UNIDAD 4
TECNOLOGÍAS PARA LA WEB

Hojas de estilos
Conceptos básicos de los estilos
Recomendación de la W3C para las hojas de estilos
Modelo de cajas
Propiedades generales de estilo para los elementos de XHTML
Propiedades avanzadas de los estilos (CSS nivel 3)

UNA BREVE CRONOLOGIA

Cuando en los albores de 1991 Tim Berneers-Lee inventó la Web, el lenguaje HTML se concibió principalmente para facilitar el intercambio de información entre los científicos de todo el mundo. Las primeras páginas web eran mucho más austeras y con un fondo gris, el texto de color negro y su estructura mucho más sencilla, con títulos y listas. ¡Fue preciso esperar varios años hasta ver aparecer las primeras imágenes... ¡en 256 colores! Estábamos bien lejos de las aplicaciones web actuales.

No obstante, muy rápidamente, en 1994, y propiciada por las relativas debilidades de HTML en términos de presentación, a la idea de agregar estilo a HTML, igual que con los programas de procesamiento de texto, apareció.

Cuando en 1995 el W3C (World Vfide Web Consortium) comenzó a funcionar, una de sus primeras tareas concretas fue la publicación, en noviembre de 1995, de un primer borrador de trabajo (working draft) acerca de las hojas de estilo. Este documento se convirtió en una recomendación oficial el 17 de diciembre de 1996 bajo el nombre de "Hojas de estilo en cascada, nivel 1" (Cascading Style Sheets/ leve/ 1) o, de forma abreviada, CSS1.

Con la emoción, se creó un grupo de trabajo específico con la misión de extender el concepto de las hojas de estilo. Se publicó una nueva recomendación el 12 de mayo de 1998 bajo el nombre de CSS2.

En 2006 se publicaron algunas correcciones y extensiones menores bajo la denominación de CSS2.1.

A finales de 2009 se comenzó a hablar de las hojas de estilo CSS3. No obstante, no podemos hablar todavía de una recomendación CSS3, puesto que el grupo de trabajo ha escindido la especificación en una multitud de módulos, cada uno siguiendo su propia evolución en el tiempo. Podemos citar el módulo CSS3 de color (CSS Color Module Leve/ 3), el módulo CSS3 de texto (¡CSS Text Leve! 3), el módulo que se ocupa de los fondos y de los bordes (¡CSS Backgrounds and Borders Leve! 3), etc.

Los distintos fabricantes de navegadores están poniendo un especial interés en estas novedades de CSS3 y las están integrando muy rápidamente. En ocasiones, incluso se implementan de forma provisional esperando a la especificación definitiva.

CONCEPTOS BÁSICOS DE LOS ESTILOS

El concepto de hoja de estilo

Las hojas de estilo son elementos agregados al lenguaje HTML que tomarán en cuenta la presentación del documento o de la aplicación web.

El concepto ligado a las hojas de estilo se sustenta en el principio de separación entre contenido y presentación a la hora de programar aplicaciones HTML. De este modo, el mismo contenido puede visualizarse, según la hoja de estilo adoptada, de forma diferente en distintos dispositivos como una pantalla tradicional, la pantalla de un ordenador de bolsillo, un teléfono móvil, una hoja de papel impreso, un intérprete braille, etc.

Así es como las hojas de estilo tienen en cuenta todo el ámbito de la presentación, limitando el rol de HTML a la estructura codificación de la información en bruto.

El CSS no es un lenguaje de programación como JavaScript ni tampoco es un lenguaje de etiquetas como HTML; de hecho, no hay nada con lo que se le pueda comparar. Las tecnologías que definían las interfaces antes del desarrollo de la web mezclaban siempre la presentación y la estructura. Sin embargo, en un entorno que cambia tan a menudo como la web, ésta no es una manera muy inteligente de hacer las cosas, y por ello se inventó el CSS.

RECOMENDACIONES DE LA W3C PARA LAS HOJAS DE ESTILOS

Resumen Este documento recoge en una definición todas las especificaciones que juntos forman el actual estado de las hojas de estilo en cascada (CSS) a partir de 2018. La audiencia principal es CSS CSS no ejecutan, autores, como esta definición incluye los módulos mediante la especificación de estabilidad, no el explorador Web índice de adopción. CSS es un lenguaje para describir la representación de documentos estructurados (como HTML y XML) en la pantalla, en papel, etc.

1. Introducción Cuando la primera especificación CSS fue publicada, todo de CSS fue contenida en un documento que define el nivel 1 de CSS. CSS de nivel 2 se define también por un único documento de varios capítulos. Pero más allá de CSS CSS Nivel 2, el Grupo de Trabajo decidió adoptar un enfoque modular, donde cada módulo define una parte de la CSS, en lugar de definir una única especificación monolítica. Esto rompe la especificación en pedazos más manejables y permite más inmediata, mejora incremental de CSS. Desde los diferentes módulos CSS están en diferentes niveles de estabilidad, el Grupo de Trabajo de CSS ha decidido publicar este perfil para definir el ámbito actual y el estado de las hojas de estilo en cascada a partir de mediados de 2018. Este perfil incluye sólo especificaciones que consideramos estable y para los cuales tenemos suficiente experiencia en implementación que estamos seguros de que la estabilidad.

Nota: Esto no pretende ser un perfil explorador de escritorio CSS: la inclusión de este perfil se basa en función de la estabilidad y no sólo el uso previsto o la adopción del navegador Web. Este perfil define CSS en su versión más completa.

Nota: Aunque no se anticipan cambios significativos en las especificaciones que forman esta instantánea, su inclusión no significa que estén congeladas. El Grupo de Trabajo continuará a abordar los problemas que se encuentran en estas especificaciones. Los implementadores deberían vigilar www-estilo y/o el grupo de trabajo de CSS Blog para eventuales cambios, correcciones o aclaraciones.

ANTECEDENTES: EL PROCESO W3C Y CSS

Esta sección no es de carácter normativo.

En el proceso del W3C, una recomendación vía el documento pasa a través de tres niveles de estabilidad, que se resumen a continuación:

1. Borrador de trabajo (WD) Esta es la fase de diseño de un W3C spec. El WG recorre la spec en respuesta a comentarios internos y externos. El primer borrador de trabajo oficial es designado como el "primer borrador de trabajo público" (FPWD). En la publicación, CSSWG FPWD indica que el Grupo de Trabajo entero ha acordado trabajar en el módulo, aproximadamente como en ámbito y propuesto en el proyecto del editor. La transición a la siguiente etapa se denomina a veces "Last Call Working Draft" (LCWD) fase. Las transiciones CSSWG borradores de trabajo una vez que se hayan resuelto todos los problemas conocidos, y puede hacer ningún progreso sin comentarios desde la construcción de pruebas e implementaciones. Esta "Última llamada para comentarios" establece un plazo para informar sobre las cuestiones pendientes, y requiere que el WG especialmente la vía y la dirección entrante de retroalimentación. El comentario del documento de seguimiento es la eliminación de comentarios (DoC). Se presentó junto con un proyecto actualizado para la aprobación del Director, para demostrar una amplia revisión y aceptación.

2. Recomendación de candidatos (CR) Esta es la fase de pruebas de una W3C spec. En particular, esta fase es sobre el uso de pruebas e implementaciones de la especificación de prueba: no se trata de probar las implementaciones. Este proceso a menudo revela más problemas con las especificaciones, y por lo tanto un candidato Recomendación va a transformarse con el tiempo en respuesta a la implementación y las pruebas de feedback, aunque generalmente menos que durante la fase de diseño (RM). Demostración de dos correctos, implementaciones independientes de cada característica es necesaria para salir de la CR, por lo que en esta fase el WG construye una suite de prueba y genera informes de ejecución. La transición a la siguiente etapa es la "Propuesta de recomendación" (PR). Durante esta fase, el Comité Asesor de W3C debe aprobar la transición a REC.
3. Recomendación (REC) Este es el estado completado de una especificación del W3C y representa una fase de mantenimiento. En este punto, el grupo de trabajo sólo se mantiene un documento de erratas y ocasionalmente publica una edición actualizada que incorpora las erratas en la especificación.

Un proyecto del Editor es efectivamente una copia viva de la copia de trabajo de los propios editores. Esto puede o no reflejar un consenso del Grupo de Trabajo, y a veces puede ser en un auto-estado incoherente. (Porque el proceso de publicación en el W3C es que consume tiempo y recursos, el proyecto del Editor es usualmente la mejor (más actualizados) para una referencia spec. En la actualidad se están realizando esfuerzos para reducir la fricción de la publicación, de modo que los borradores oficiales será regularmente actualizada y Editor's borradores pueden volver a su función original como espacio de desecho).

LOS NIVELES DE CSS

Las hojas de estilo en cascada no tienen versiones en el sentido tradicional sino que tiene niveles. Cada nivel de CSS se basa en el anterior, el refinamiento de las definiciones y la adición de características. El conjunto de características de cada nivel superior es un super conjunto de cualquier nivel inferior, y el comportamiento permitido para una función determinada en un nivel superior es un subconjunto de lo permitido en los niveles inferiores. Un agente de usuario conforme a un mayor nivel de CSS es, por tanto, también cumple a todos los niveles inferiores.

CSS NIVEL 1

El Grupo de Trabajo considera que la especificación CSS1 a ser obsoleta. CSS de nivel 1 se define como todas las características definidas en la especificación CSS1 (propiedades, valores, normas, etc), pero con la sintaxis y las definiciones en la especificación CSS2.1. Los atributos de estilo CSS define su inclusión en el elemento específico de atributos de estilo.

CSS NIVEL 2

Aunque el nivel 2 de CSS CSS2 especificación es técnicamente una recomendación de W3C, pasa a la etapa de recomendación antes de la W3C ha definido la etapa de recomendación de candidato. A lo largo del tiempo la experiencia de implementación y revisión ulterior ha sacado a la luz muchos de los problemas de la especificación CSS2, así que en lugar de ampliar la lista de erratas ya suficientemente rígidos, el Grupo de Trabajo eligió CSS para definir CSS Nivel 2 Revisión 1 (CSS2.1). En caso de cualquier conflicto entre las dos especificaciones CSS2.1 contiene la definición definitiva. Una vez CSS2.1 se convirtió en candidato Recomendación eficaz, aunque no oficialmente el mismo nivel de estabilidad como CSS2-obsoleta la Recomendación CSS2.

Características de CSS2 que fueron lanzadas desde CSS2.1 deben considerarse en la fase de Recomendación de candidato, pero tenga en cuenta que muchos de estos han sido o van a ser arrastrados a una CSS Nivel 3 Borrador de trabajo, en cuyo caso dicha especificación, una vez que alcanza la CR, obsoletas las definiciones en CSS2.

La especificación CSS2.1 define el nivel 2 de CSS y los atributos de estilo CSS especificación define su inclusión en el elemento específico de atributos de estilo.

NIVEL 3 DE CSS

CSS de nivel 3 se basa en CSS Nivel 2 módulo por módulo, utilizando la especificación CSS2.1 como su núcleo. Cada módulo agrega funcionalidad y/o reemplaza parte de la especificación CSS2.1. El Grupo de Trabajo de CSS se propone que los nuevos módulos CSS no contradicen la especificación CSS2.1: sólo que le agregan funcionalidad y afinar las definiciones. Como cada módulo esté terminado, será conectado al sistema existente de CSS2.1 plus módulos terminados previamente. A partir de este nivel los módulos se nivelan de forma independiente: por ejemplo selectores Nivel 4 podría completarse antes de Módulo de línea CSS Nivel 3. Los módulos con ningún equivalente de nivel 2 de CSS empieza en el nivel 1; módulos que actualizar características que existía en CSS Nivel 2 empieza en el nivel 3.

CSS NIVEL 4 Y MÁS ALLÁ NO HAY

CSS Nivel 4. Los módulos independientes pueden alcanzar el nivel 4 o más allá, pero el lenguaje CSS ya no tiene niveles. (“CSS Nivel 3” como un término que se utiliza sólo para diferenciarla de las anteriores versiones monolíticas.)

PERFILES DE CSS

Los perfiles de todas las implementaciones de CSS no implementará todas las funciones definidas en CSS. En el pasado, el Grupo de Trabajo publicó algunos perfiles, que eran para definir el subconjunto mínimo de CSS que diversas clases de agentes de usuario se espera apoyar. Este esfuerzo ha sido suspendido, como el Grupo de Trabajo no estaba resultando eficaz o útil, y los perfiles definidos previamente son ahora no mantenido.

- Nota: las implementaciones parciales de CSS, incluso si ese subconjunto es un perfil oficial, deben seguir las reglas de análisis compatible para las implementaciones parciales.

REQUISITOS PARA LA APLICACIÓN RESPONSABLE DE CSS

En las siguientes secciones se definen varios requisitos de conformidad para aplicar CSS responsablemente, de una manera que promueva la interoperabilidad en el presente y el futuro.

- Implementaciones parciales

Las implementaciones parciales para que los autores puedan explotar la compatible analizar reglas para asignar valores de suplencia, CSS representadores debe tratar como no válido (según corresponda) e ignorar cualquier en normas, propiedades, valores de propiedades, palabras clave y otras construcciones sintácticas para las cuales no tienen ningún nivel de soporte utilizable.

En particular, los agentes de usuario no deben ignorar selectivamente los valores de propiedad no admitido y honrar los valores admitidos en una sola declaración de propiedad de valor múltiple: si ningún valor es considerada no válida (como valores no admitidos deben estar), CSS requiere que toda la declaración será ignorado.

- Implementaciones de inestables y las funciones exclusivas

Para evitar enfrentamientos con características de CSS estables en el futuro, la CSSWG recomienda las siguientes prácticas recomendadas para la implementación de características inestables y extensiones propietarias de CSS:

- Experimentación y características inestable

Las implementaciones inestables de características que se describen en las especificaciones de W3C pero no son interoperables no deberían ser liberados ampliamente para uso general; pero puede ser liberado para uso experimental, limitado en entornos controlados. Porque?

Una característica de CSS se consideraba inestable hasta su especificación ha alcanzado la recomendación de candidatos (CR) etapa de W3C en el proceso. En casos excepcionales, el CSSWG podrán además, por resolución registrada oficialmente, añadir pre-CR características al conjunto que se consideran seguros para liberar de amplio uso.

PATENTADO Y CARACTERÍSTICAS NO ESTANDARIZADAS

Para evitar enfrentamientos con futuras características CSS, la especificación CSS2.1 reserva un prefijo sintaxis [CSS2] de especialidades y extensiones experimentales a CSS. Una característica de CSS es una extensión propiedad si está pensada para el uso en un entorno cerrado accesible sólo a un único proveedor de agente de usuario(s). Un UA debe apoyar tales extensiones propietarias sólo a través de un proveedor de sintaxis prefijado y no exponerlos a abrir (multi-UA) entornos tales como el World Wide Web.

Incluso si una característica está destinada a llegar a ser utilizado en la Web, si aún no se ha normalizado todavía no debe estar expuesta a la Web.

LA PRESIÓN DE LOS MERCADOS Y DE LAS NORMAS DE FACTO

Si una función es inestable (es decir, la especificación no se ha estabilizado aún), pero

Al menos tres UAs implementar la característica (o un UA ha roto las demás reglas y transportado por el amplio uso una inestable o no-característica estándar en una versión de producción),

Y las implementaciones Interoperabilidad irregular,

Y el Grupo de Trabajo de CSS ha registrado un consenso que esta función debe existir y ser liberado,

IMPLEMENTACIONES DE CARACTERÍSTICAS DE NIVEL DE CR

Una vez que una especificación llega a la etapa de recomendación de candidato, los implementadores deben liberar una aplicación unprefixed de cualquier característica de nivel CR pueden demostrar ser implementado correctamente según las especificaciones, y deben evitar la exposición a un prefijo variante de esa característica.

Para establecer y mantener la interoperabilidad a través de implementaciones de CSS CSS, el Grupo de Trabajo pide que no experimental representadores CSS presentara un informe de implementación (y, si es necesario, los testcases utilizados para que la aplicación informe) al W3C antes de soltar un unprefixed la aplicación de cualquiera de las características de CSS. Testcases presentadas al W3C están sujetas a revisión y corrección por parte del Grupo de Trabajo de CSS.

Más información sobre la presentación de informes de ejecución y testcases pueden encontrarse en el sitio web del Grupo de Trabajo de CSS en https://www.w3.org/Style/CSS/Test/. Las preguntas deben ser dirigidas a la Public-css-testsuite@w3.org lista de correo.

SEGURO DE LIBERAR EXCEPCIONES A PRE-CR

Las siguientes características han sido explícitamente y borrado de forma proactiva por el Grupo de Trabajo de CSS antes de la liberación de amplio alcance spec candidato Recomendación:

- El aporte relativo de las propiedades equivalentes de tamaño (anchura, altura, etc.), las propiedades de borde, las **propiedades margin** y **padding**. Véase la explicación y la especificación.
- El contenido mín y máx-content palabras clave de las propiedades de tamaño. Véase la decisión y especificación.
- El **CONIC-gradient()** notación de degradado. Véase la decisión.

Las siguientes características han sido explícitamente y borra retroactivamente por el Grupo de Trabajo de CSS antes de la liberación de amplio alcance spec candidato recomendación:

- Todo en la CSS animaciones, transiciones de CSS de nivel 1, Nivel 1 y Nivel 1 Transformaciones CSS.
- La **:dir(), :lang()** y **:focus** dentro de pseudo-classes de selectores [4].
- Para mayor informacion visite la pagina de la w3c:
- https://www.w3.org/TR/css-2018/

EL MODELO DE CAJAS DE CSS

Todos los elementos de HTML se consideran una "caja", ya se trate de un párrafo, un , una imagen, etc. Las cajas poseen propiedades consistentes, con inde pendencia de que las veamos o no, y de que se especifiquen en la hoja de estilo o no. Siempre están presentes, y como diseñadores, debemos tenerlas presentes al crear un diseño.

La figura siguiente es un diagrama del modelo de cajas. Este modelo describe cómo cada elemento de bloque de HTML tiene la posibilidad de tener un borde, un margen y un relleno, y cómo se aplicarían estos. En otras palabras, todos los elementos tienen algo de relleno en el contenido y el borde del elemento. Además, el borde podría ser visible o no, aunque haya espacio para ello, del mismo modo en que hay un margen entre el borde del elemento y cualquier otro contenido externo al elemento.

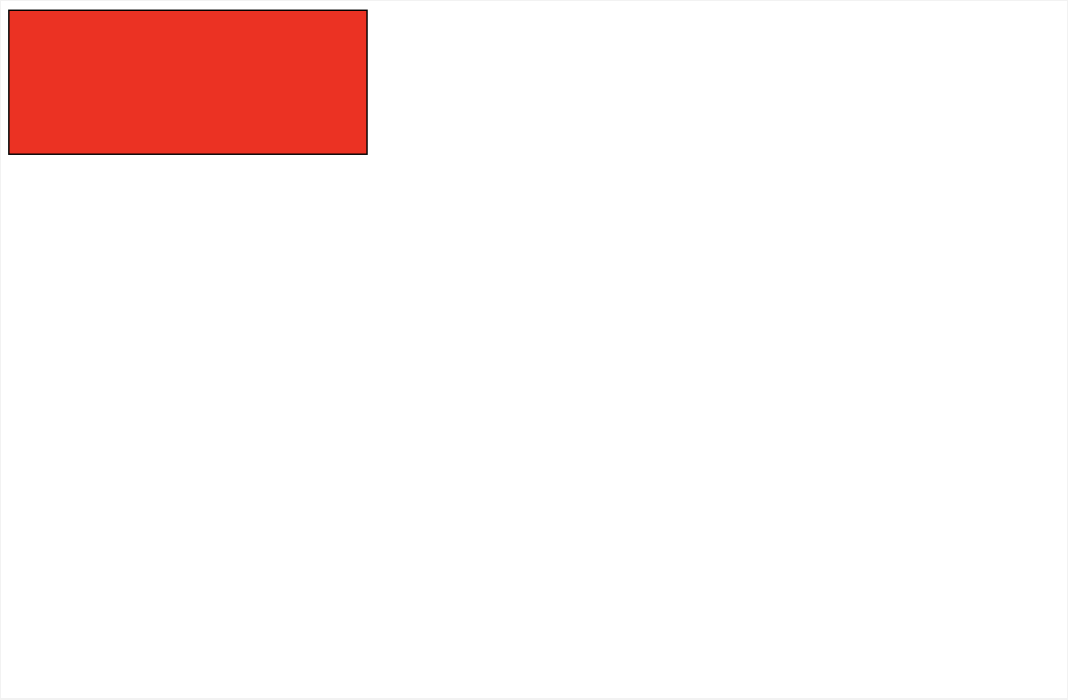
Veamos otra explicación más del modelo de cajas, yendo de fuera adentro:

- El margen es el área externa al elemento. Nunca tiene color, es siempre transparente.
- El borde se extiende alrededor del elemento, por el límite externo del posible relleno. El borde puede ser de varios tipos, anchos y colores.
- El relleno existe alrededor del contenido y hereda el color de fondo de su área.
- El contenido está rodeado por el relleno.

Ahora viene la parte complicada:

Para saber la altura y la anchura reales de un elemento, deberá tener en cuenta todos los elementos del modelo de cajas. Si recuerda el ejemplo del capítulo anterior, en el que a pesar de indicar que un debía tener 250 pixeles de ancho y 100 píxeles de alto, tuvo que ampliarse para hacer sitio al relleno utilizado. Ya sabe cómo definir la anchura y la altura de un elemento usando las propiedades width y height . El siguiente ejemplo muestra cómo se define un que tiene 250 píxeles de ancho, 100 de alto, un fondo rojo y un borde negro de un píxel:

```
<!DOCTYPE html>
<html>
<head>
     <meta charset="UTF-8">
     <meta name="autor" content="Los macheteros">
     <link rel="stylesheet" type="text/css" href="estilos.css">
</head>
<link href="estilos.css" rel="stylesheet" type="text/css">
<body>
     <div class="div1">
     </div>
</body>
</html>
```



Si definimos un segundo elemento con estas mismas propiedades, pero añadimos también unas propiedades margin y padding de un determinado tamaño, empezaremos a ver cómo cambia el tamaño del elemento. Esto se debe al modelo de cajas.

El segundo se definirá como vemos a continuación, añadiendo solo 10 píxeles de margen y 10 píxeles de relleno al elemento:

El segundo , que vemos en la figura se define con la mismas altura y anchura que el primero, pero la altura y la anchura globales de toda la caja que rodea al elemento son mucho mayores cuando entran en juego los márgenes y los rellenos.

La anchura total de un elemento es la suma de lo siguiente:

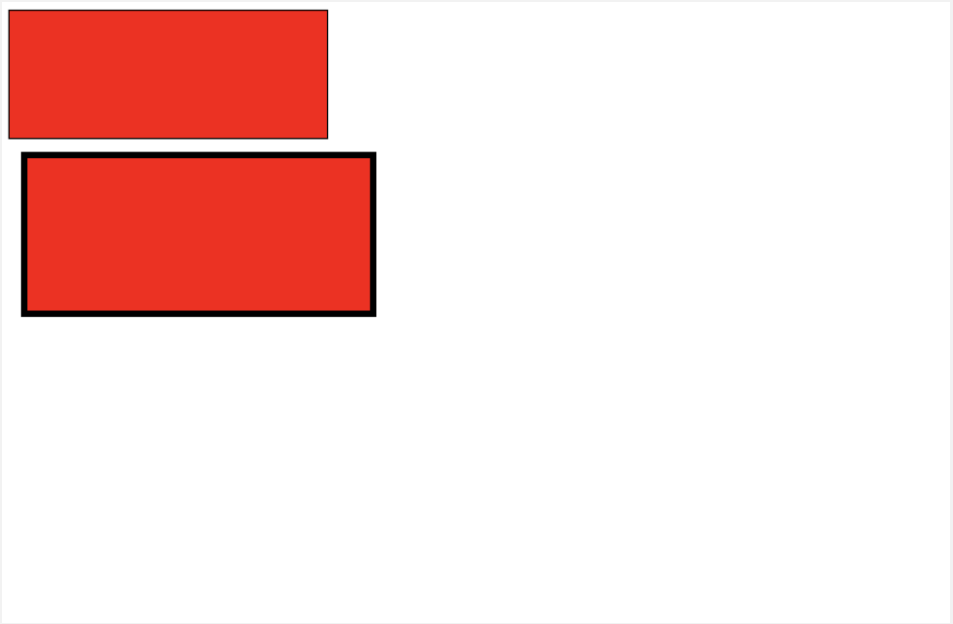
width + padding-left + padding-right + border-left + border-right + margin-left + margin-right

La altura total de un elemento es la suma de lo siguiente:

height + padding-top+ padding-bottom + border-top + border-bottom + margin-top + margin-bottom

Por tanto, la anchura real del segundo es de (250 +10+10+5 +5 +10+10) y su altura real es de 150 (100 + 10 +10 + 5 + 5 + 10 +10). Como puede observar, el modelo de cajas afecta al diseño. Supongamos que solo dispone de 250 píxeles de espacio horizontal y desea incluir 10 píxeles de margen, 10 de relleno y 5 de borde en todos los lados. Para conciliar estos parámetros con las posibilidades que ofrece el espacio horizontal, tendrá que especificar un ancho para el de solo 200 píxeles, de modo que al sumar 200 + 10 + 10 + 5 + 5 + 10 + 10 obtengan los 250 píxeles del espacio horizontal disponible.

```html
<!DOCTYPE html>
<html>
<head>
     <meta charset="UTF-8">
     <meta name="autor" content="Los macheteros">
     <link rel="stylesheet" type="text/css" href="estilos.css">
     </head>
<link href="estilos.css" rel="stylesheet" type="text/css">
<body>
     <div class="div1">
     </div>
     <div class="div2">
     </div>
</body>
</html>
```



Aparte de entender el concepto que hay detrás del modelo de cajas, es importante que conozca también los cálculos asociados al modelo. En los sitios de generación dinámica
o en aquellos en los que la visualización en el lado del cliente depende de la interacción del usuario (como los basados en eventos de JavaScript), el código del lado del servidor
o del lado del cliente puede dibujar o redibujar elementos contenedores sobre la marcha. En otras palabras, su código producirá los números, pero usted tendrá que proporcionar
los límites.

Ahora que ya sabe cómo funciona el modelo de cajas, téngalo presente durante el resto de los trabajos que realice y en sus diseños para la Web. Entre otras cosas, afectará al posicionamiento de elementos y al flujo de los contenidos, que son los dos asuntos que vamos a abordar a continuación.

TODO SOBRE EL POSICIONAMIENTO.

El posicionamiento relativo es el tipo de posicionamiento por defecto que se utiliza en HTML. Puede concebirse como algo similar a la disposición de las damas en un tablero: Se colocan de izquierda a derecha, y al llegar al límite del tablero, se pasa a la siguiente fila. Los elementos cuyo estilo incluye el valor block para la propiedad display se colocan automáticamente en una nueva fila, mientras que los elementos inline se colocan en la misma fila, justo a continuación del elemento que los precede. Por ejemplo, las etiquetas <p> y <di v> se consideran elementos de bloque, mientras que la etiqueta se considera un elemento en línea.

El otro tipo de posicionamiento que soporta CSS se denomina posicionamiento absoluto, porque le permite definir la posición exacta del contenido HTML de una página. Aunque el posicionamiento absoluto le da la libertad de indicar exactamente dónde se ubicará un elemento, la posición sigue siendo relativa a todos los elementos padre que aparecen en la página. En otras palabras, el posicionamiento absoluto le permite especificar la ubicación exacta del área rectangular de un elemento con respecto al área de su padre, algo muy diferente al posicionamiento relativo.

Esta libertad para colocar los elementos en el lugar que desee de la página le puede conducir al problema del solapamiento, que tiene lugar cuando un elemento ocupa el espacio usado por otro elemento. No hay nada que le impida especificar posiciones absolutas de elementos de modo que se solapen. En este caso, CSS emplea el índice z de cada elemento para determinar qué elemento está encima y cuál está debajo. Aprenderá más sobre el índice z de los elementos más adelante en este capítulo. De momento, vamos a ver cómo controlar exactamente si una regla de estilo utilizará un posicionamiento absoluto o relativo.

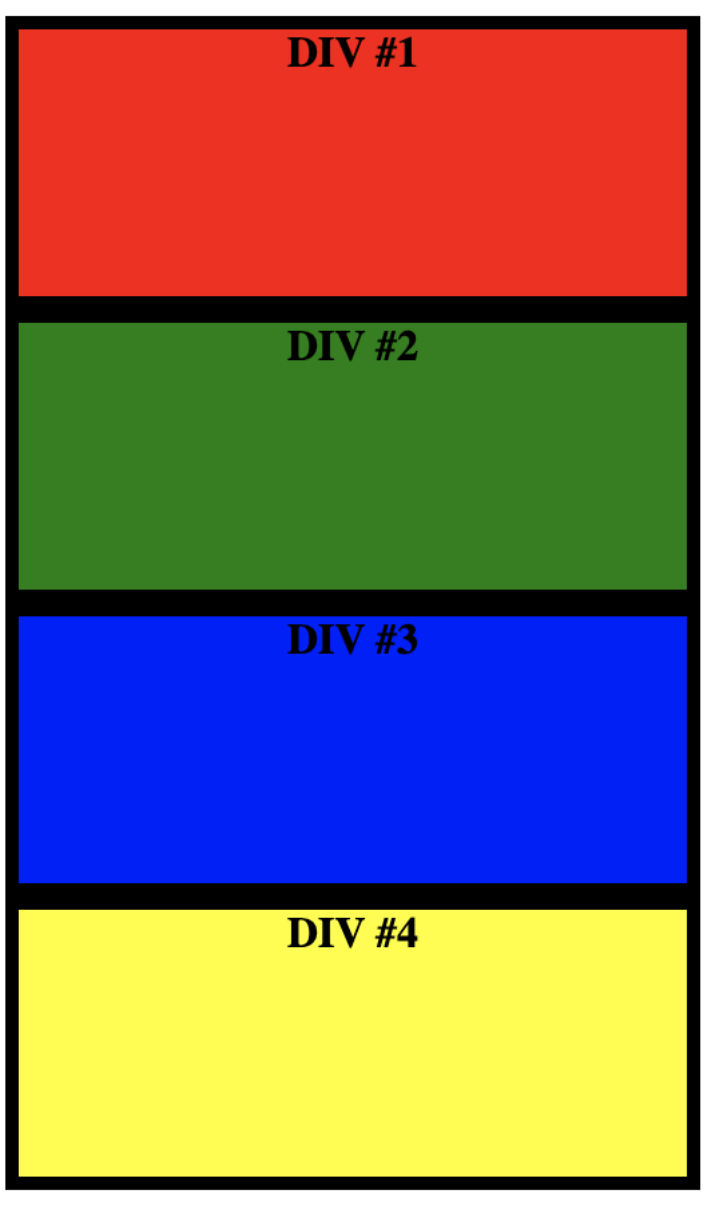
El tipo de posicionamiento (relativo o absoluto) usado por una regla de estilo particular viene determinada por la propiedad position, que puede tener uno de estos dos valores: relativeoabsolute. Después de especificar el tipo de posicionamiento, se proporciona la posición específica utilizando las siguientes propiedades:

- left: El desplazamiento de la posición izquierda.
- right: El desplazamiento de la posición derecha.
- top: El desplazamiento de la posición superior.
- bottom: El desplazamiento de la posición inferior.

¿Quizá piense que estas propiedades de las posiciones sol? tienen sentido en el posicionamiento absoluto, pero en realidad se aplican en ambos tipos. Dentro del posicionamiento relativo, la posición de un elemento se especifica como un desplazamiento relativo a la posición original del elemento. Así pues, se define la propiedad left de un elemento con el valor 25px, el lado izquierdo de éste se vera desplazado 25 pixeles de su posición original (relativa). Por el contrario, una posición absoluta especifica en relación con el padre del elemento al que se aplica dicho estilo. De este modo, si define la propiedad left de un elemento con el valor 25px dentro del posicionamiento absoluto, el lado izquierdo del elemento se mostrará 25 píxeles a la derecha del extremo izquierdo de su padre. Por el contrario, al usar la propiedad right con el mismo valor, el elemento se situaría de manera que su lado derecho estuviese a 25 píxeles del extremo derecho del padre.

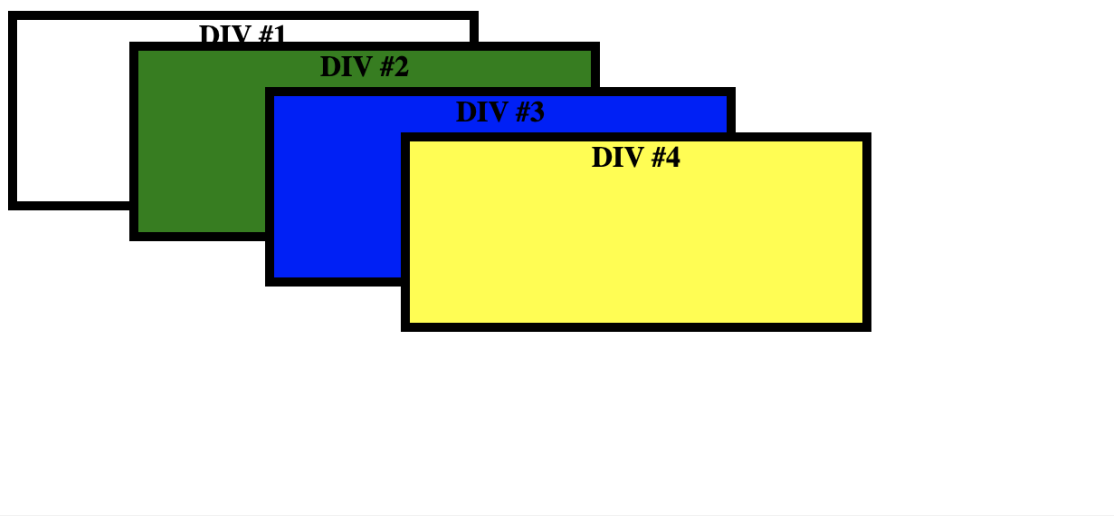
Vamos a volver al ejemplo de los bloques de colores para mostrar cómo funciona el posicionamiento. En el listado se especifica un posicionamiento relativo para los cuatro bloques. Como puede observar en la figura los bloques se distribuyen verticalmente.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xh
tmlll/DTD/xhtmlll.d td"">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
     <title>Positioning the Color Blocks</title>
     <style type="text/css">
          div {
          position:relative;
          width:250px;
          height:100px;
          border:5px solid #000;
          color:black;
          font-weight:bold;
          text-align:center;
          }
          #d1{
          background-color:red;
          }
          #d2{
          background-color:green;
          }
          #d3{
          background-color:blue;
          }
          #d4{
          background-color:yellow;
          }
</style>
</head>
<body>
     <div id="d1">DIV #1</div>
     <div id="d2">DIV #2</div>
     <div id="d3">DIV #3</div>
     <div id="d4">DIV #4</div>
</body>
</html>
```



La entrada común de la hoja de estilo para elelemento define como relativo en la propiedad de estilo position del elemento . Como las reglas de estilo restantes heredan de dicha regla, también heredan su posicionamiento relativo. De hecho, la única diferencia con las demás reglas de estilo es que poseen distintos colores de fondo. En la figura se puede observar que los elementos se colocan unos tras otros, que es lo esperado cuando se emplea posicionamiento relativo. Vamos a transformar esto en algo más interesante, cambiando el posicionamiento a absoluto y especificando explícitamente la ubicación de los colores. En el siguiente se han cambiado las entradas de la hoja de estilo para que utilicen un posicionamiento absoluto en la colocación de los bloques de colores.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11 .dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
        <title>Positioning the Color Blocks</title>
        <style type="text/css">
                div {
                position:absolute;
                width:250px;
                height:100px;
                border:5px solid #000;
                color:black;
                font-weight:bold;
                text-align:center;
                }
                #d1{
                background-color:red;
                left:0px;
                top: 0px;
                }
                #d2{
                background-color:green; left:75px;
                top:25px;
                }
                #d3{
                background-color:blue;
                left:150px;
                top:50px;
                }
                #d4{
                background-color:yellow; left:225px;
                top:75px;
                }
</style>
</head>
<body>
        <div id="d1">DIV #1 </div>
        <div id="d2">DIV #2 </div>
        <div id="d3">DIV #3 </div>
        <div id="d4">DIV #4 </div>
</body>
</html>
```



Esta hoja de estilo define la propiedad position como absolute, que es lo que hace que la hoja de estilo emplee un posicionamiento absoluto. Además, en todas las reglas de estilo heredadas se han definido las propiedades left y top. Sin embargo, la posición de cada una de estas reglas se ha definido de manera que los elementos se muestren solapándose entre sí, como muestra la figura. Bueno, pues ya estamos hablando de estructuras. La figura muestra cómo puede utilizar el posicionamiento absoluto para colocar los elementos exactamente como desee. También revela lo fácil que es colocar los elementos para que se solapen entre sí. Puede que tenga curiosidad por saber cómo sabe un navegador qué elementos debe dibujar encima de otros cuando estos se solapan. En la siguiente sección veremos cómo se puede controlar el orden de apilamiento.

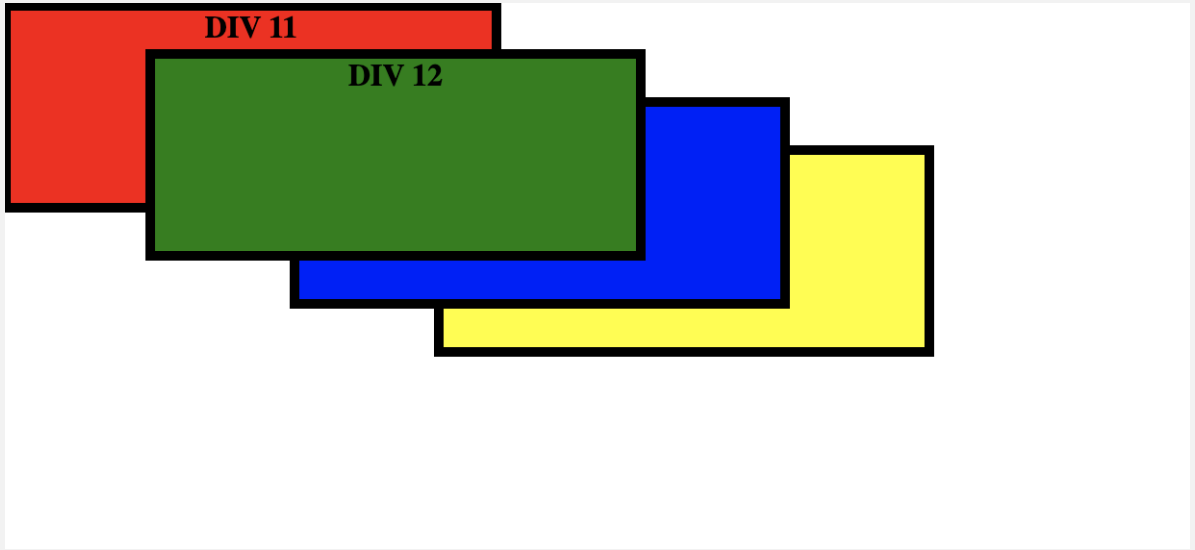
CONTROLAR COMO SE APILAN LAS COSAS.

Habrá situaciones en las que necesite tener un control meticuloso del modo en que se solapan los elementos entre sí en una página Web. La propiedad de estilo z-index le permite establecer el orden de los elementos en lo relativo a cómo se apilan unos sobre otros. Aunque el nombre z-index quizá le resulte un poco extraño, hace referencia al concepto de una tercera dimensión (Z) que apunta hacia la pantalla delordenador, y que se suma a las otras dos dimensiones que recorren la pantalla a lo ancho (X) y a lo alto (Y). Otro modo de concebir este índice z es la posición de una única revista en una pila de revistas. Una revista que esté más cerca de la parte superior del montón tendrá un índice z superior al de otra que se encuentre más abajo. Del mismo modo, un elemento solapado con un índice z elevado se mostrará sobre otro con un índice z inferior.

La propiedad z-index se emplea para asignar un valor numérico que indica el índice z relativo de una regla de estilo. El número asignado a z-index solo tiene significado con respecto las demás reglas de estilo de una hoja de estilo, de modo que definir esta propiedad en una única regla no tiene demasiado sentido. Por el contrario, si define z-index para varias reglas de estilo que se aplican a elementos solapados, loselementos con valores z-index superiores se mostrarán por encima de los elementos con valores inferiores. El listado 10.3 contiene otra versión de la hoja de estilos de los bloques de colores y un código HTML que emplea parámetros de z-index para alterar el solapamiento
natural de los elementos.

Listado 10.3. Uso de z-index para alterar la visualización de los elementos en el ejemplo de los bloques.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xh
tmlll/DTD/xhtmlll.d td"">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
	<title>Positioning the Color Blocks</title>
	<style type="text/css">
		div{
		position:absolute;
		width:250px;
		height:100px;
		border:5px solid #000;
		color:black;
		font-weight:bold;
		text-align:center;
		}
		#d1{
		background-color:red;
		left:0px;
		top:0px;
		z-index:0;
		}
		#d2{
		background-color:green; left:75px;
		top:25px;
		z-index:3;
		}
		#d3{
		background-color:blue;
		left:150px;
		top:50px;
		z-index:2;
		}
		#d4{
		background-color:yellow;
		left:225px;
		top:75px;
		z-index:1;
		}
</style>
</head>
<body>
	<div id="d1">DIV 11</div>
	<div id="d2">DIV 12</div>
	<div id="d3">DIV 13 </div>
	<div id="d4">DIV 14 </div>
</body>
</html>
```



El único cambio en este código respecto al listado 10.2 es la incorporación de la propiedad z-index en cada una de las clases div numeradas. Observe que el primer div numerado tiene un z-index de valor 0, lo que lo convierte en el elemento de nivel de índice z más bajo, mientras que el segundo div posee el z-index más alto. La figura 10.6 muestra cómo se vería la página de los bloques con esta hoja de estilo, donde podemos observar claramente cómo afecta el z-index a la visualización del contenido y permite controlar con precisión el solapamiento de los elementos.

Aunque los ejemplos muestran bloques de color que son simples elementos , la propiedad z-index puede afectar a cualquier tipo de contenido HTML, imágenes incluidas.

CONTROLAR EL FLUJO DEL TEXTO

Ahora que ya ha visto algunos ejemplos de la colocación de los elementos en relación a otros o de la ubicación absoluta de estos, llega el momento de repasar cómo fluyen los contenidos alrededor de los elementos. El concepto de línea actual es una línea invisible que se usa para colocar los elementos en una página. Esta línea está relacionada con el flujo de los elementos en una página; entra en juego conforme los elementos se van distribuyendo los unos junto a los otros a lo largo y ancho de la página. Parte del flujo de los elementos es el flujo del texto de una página. Cuando se mezclan textos con otros elementos (como las imágenes), es importante controlar cómo fluye el texto alrededor de estos otros elementos.

Ya ha visto dos de estas propiedades de estilo. A continuación vamos a ver algunas propiedades de estilo que le proporcionarán control sobre el flujo del texto:

- float: Determina cómo fluye el texto alrededor de un elemento.
- clear: Detiene el flujo del texto alrededor de un elemento.
- overflow: Controla el desbordamiento del texto cuando un elemento es demasiado pequeño para contener todo el texto.

La propiedad float se utiliza para controlar cómo fluye el texto alrededor de un elemento. Puede tomar los valores left o right. Estos valores determinan dónde se colocará un elemento con respecto al texto que fluye. Por tanto, si asignamos el valor left a la propiedad float de una imagen, ésta se colocará a la izquierda del texto. Como vimos en el capítulo anterior, se puede impedir que el texto fluya junto a un elemento empleando la propiedad clear, que puede tomar los valores none, left, right o both.

El valor por defecto de la propiedad clear es none, queindica que el texto fluye junto al elemento sin ninguna consideración en especial. El valor left hace que el texto deje de flotar alrededor de un elemento hasta que el lado izquierdo de la página quede libre de dicho elemento. Del mismo modo, el valor right haceque el texto no fluya por el lado derecho del elemento. El valor both indica que el texto no flotará a ningún lado del elemento.

La propiedad overflow controla el texto desbordado, que es aquel que no cabe dentro de su área rectangular. Esto puede ocurrir si se definen valores demasiado pequeños para el width yel height deun elemento. La propiedad overflow se puede definir como visible, hidden o scroll. El valor visible agranda automáticamente el elemento para que quepa todo el texto dentro de él; éste es el valor por defecto de la propiedad. El valor hidden deja al elemento con el mismo tamaño, pero oculta a la vista la parte del texto que no cabe. El valor más interesante probablemente sea scroll , que añadebarras de desplazamientoal elemento para poder desplazar el texto y leerlo completo.

PROPIEDADES AVANZADAS PARA LOS ELEMENTOS CSS 3 NIVEL 3

Los bordes

- **Borde con colores diferentes**

La propiedad -moz-border-colors: permite crear varios bordes de colores diferentes. Esta propiedad puede ser utilizada también como: -moz-borders-top-color: (adición de bordes, top, bottom, left, right)

#midiv{border: 8px solid #000000;moz-border-colors:#CC0000 #BB0000 #AA0000 #990000 #880000 #7700000 #660000 #550000; padding: 5px;}

En imagen:

- ***Imágenes como bordes***

CSS3 permite el uso de imágenes como bordes de los elementos de la página.

Las dos propiedades

- border-image
- border-top-image
- border-right-image
- border-bottom-image
- border-left-image
- border-corner-image
- border-top-left-image
- border-top-right-image
- border-bottom-left-image
- border-bottom-right-image
- *#mi-div{border-image: url (border.png) 27 27 27 27 round round;}*

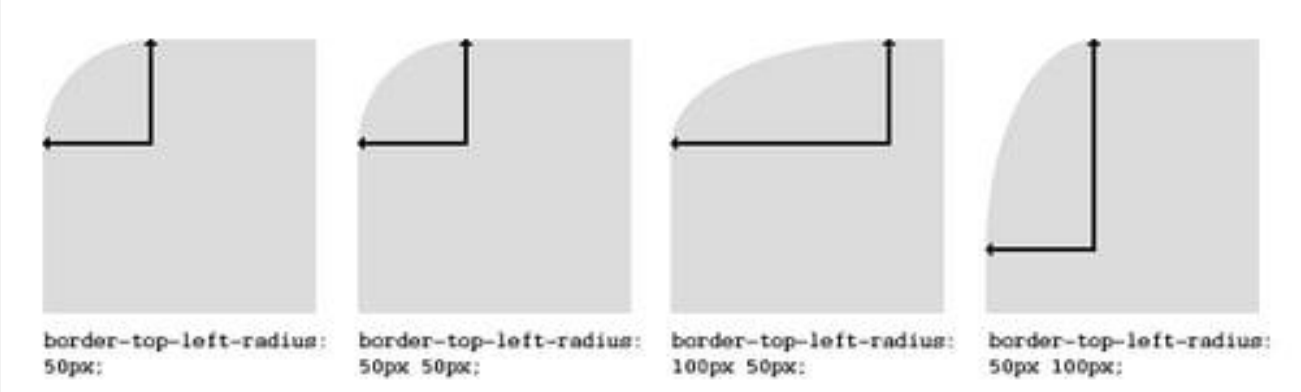
En imagen:

#mi_div{border-iamge: url(border.png) 27 27 27 27 stretch stretch}

Bordes redondeados en las esquinas

La propiedad <gras<border-radius</gras> de CSS3 permite a los desarrolladores web definir bordes redondeados en las esquinas, sin necesidad de imágenes de esquinas ni recurrir al uso de etiquetas div multiples.

#contenedor{-moz-border-radius: 15px;border-radius: 15px;}



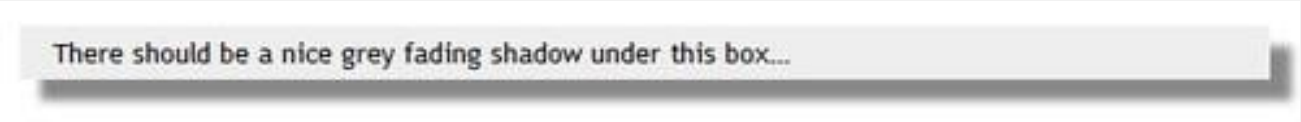
Una nueva funcionalidad de CSS3 implementada a partir de la versión 3.1 de Firefox, es la posibilidad de crear sombras de colores: esta es la propiedad box-shadow.

Esta propiedad requiere de algunos parámetros para definir las características de la sombra:

1. Desplazamiento horizontal de la sombra: un valor positivo significa que la sombra aparece desde la derecha, un desplazamiento negativo hará que la sombra aparezca desde la izquierda.
2. Desplazamiento vertical de la sombra: un valor negativo significa que el box-shadow aparecerá desde arriba, un valor negativo hará aparecer la sombra desde abajo.
3. En cuanto al difuminado, cuanto más cerca de cero esté este valor, la sombra será más definida. En cambio, cuanto más se acerque de 1, la sombra estará más difuminada.

Ejemplos:

Sombra{box-shadow: 10px 10px 5px #888; padding: 5px 5px 5px 15px;}

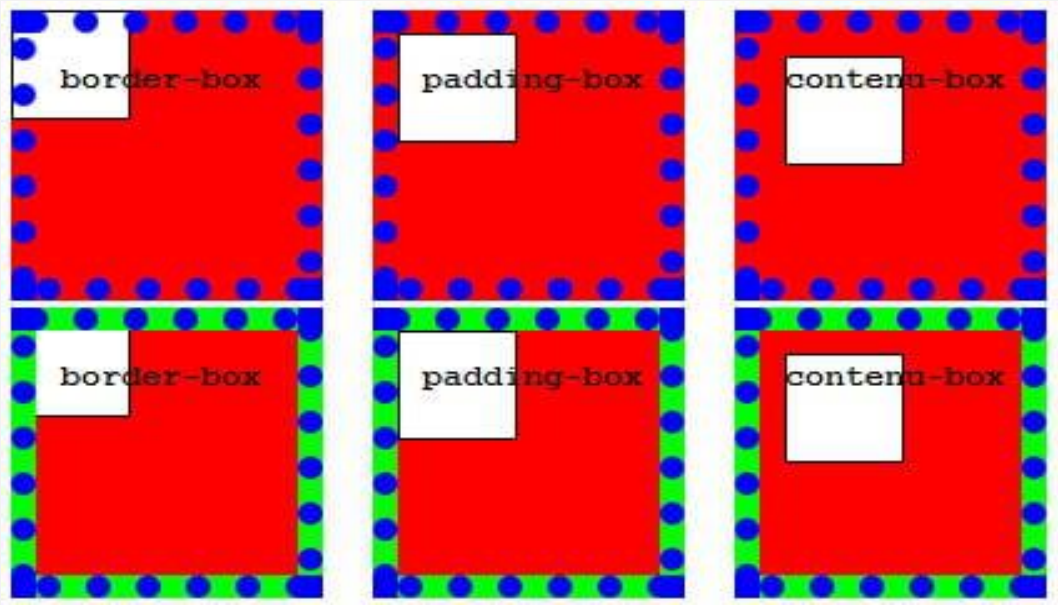


Sombra_2{box-shadow:-10px-10px 0px #000000; border-radius: 5px; padding: 5 px 5px 5px 15px; }

IMÁGENES DE FONDO DE ELEMENTOS EN CSS3

- Background-clip y Background-origin

La propiedad background-origin de CSS3 permite determinar la manera en que la imagen de fondo se posicionará en un elemento. Esta propiedad toma 3 valores: border-box, padding-box y content-box.



BACKGROUND-SIZE

CSS3 permite especificar un tamaño a las imágenes de fondo. Este tamaño puede ser especificado en pixeles, (height y width), o en porcentaje. Si se especifica una tamaño en porcentaje, el tamaño es relativo al ancho o altura de la zona a la que se ha atribuido la propiedad background-origin.

Imágenes de fondo multiples en CSS3

Es simple:

Background: *url(body-top.gif) topleft no repeart, url(banner_freco.jpg) top 1 1px no-repeat, ur(body-bottom.gif) bottom left no-repeat,url(body-middle.gif)left repeat;*

Nota: estas son propiedades experimentales

LOS COLORES EN CSS3

CSS3 podria ver la introducción de la propiedad HSL (Matiz, Saturacion, Luminosidad). HSL toma tres valores:

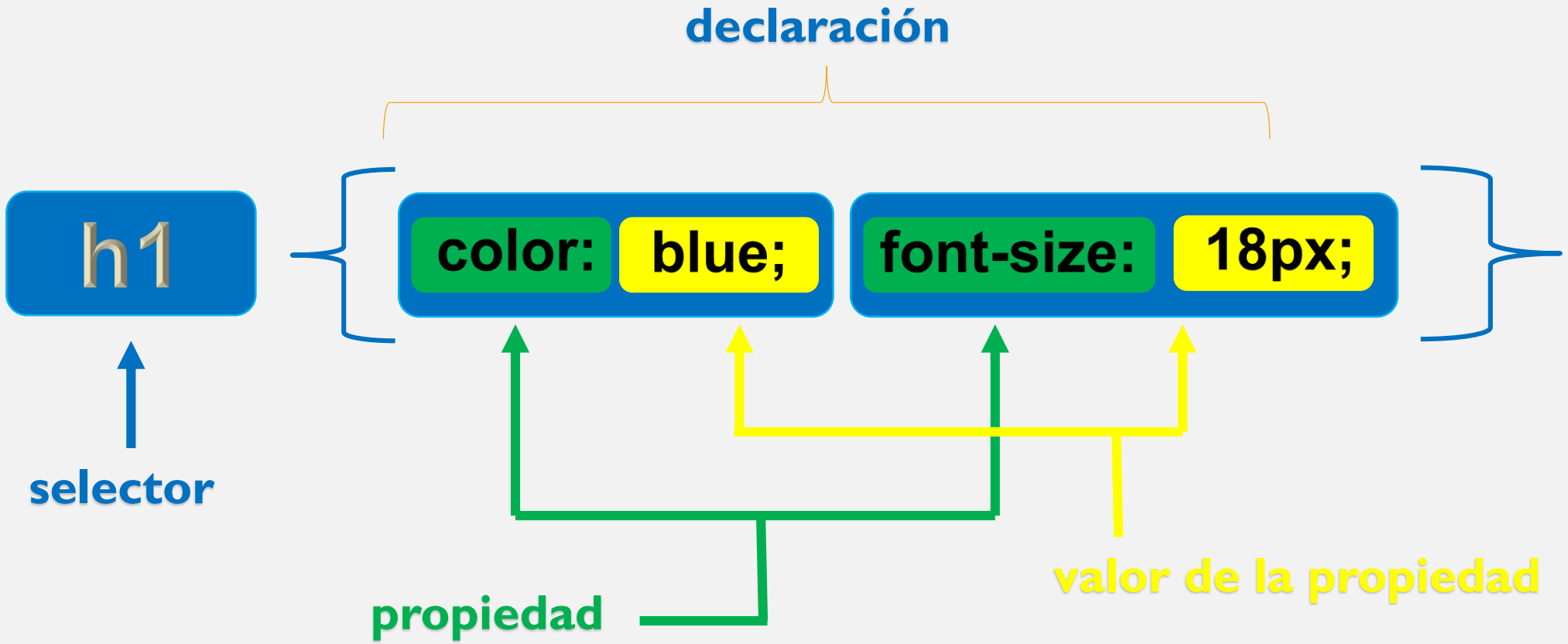
- Matiz: corresponde a la tinta: 0 (0 360) es de color rojo, verde vale 120 y, 240 es azul. Los otros valores son matices de colores.
- Saturación:La saturación es un valor en porcentaje. 100% es el color exacto.
- Luminosidad: La luz es igualmente un porcentaje. 0% es la sombra (negro), y 100% completamente claro (blanco), 50% corresponde al valor medio.

Esto da una amplia gama de posibilidades en cuanto al tono de los colores. Hasta el momento, HSL ha sido implementado en Opera 9.5, safari 3, Konqueror y Mozilla.

Ejemplo:

#mid_div_hsl{background-color:hsl(240,100%,50%);}

CSS SINTAXIS



SELECTOR

- Es Importante hacer el marcado de contenido del sitio Web con etiquetas html, para poder aplicar el lenguaje CSS para que se encargue de decorar y dar estilo a las páginas web.

- Es importante tener una comprensión sintáctica mínima acerca de lo que es el lenguaje css, es decir aplicar la sintaxis para poder aprovechar las funcionalidades que se le puede dar

SELECTOR

- un selector es una especie de condición que definimos dentro de la hoja de estilos para lograr seleccionar uno o varios elementos ubicados dentro del marcado html.
- Deben de tener en cuenta qué se debe de cumplir las condiciones para cada selector.
- una hoja de estilos contiene un conjunto de selectores con sus declaraciones respectivas de estilos al lado es decir indicando las propiedades a las que se le van a hacer modificaciones o cambios

SELECTORES DE ELEMENTOS DEL LENGUAJE HTML

Selecciona todos los elementos definidos en el lenguaje HTML independientemente de la versión que se ocupe, lo importante como bien se menciona en diapositivas anteriores es aplicar bien la condición, es decir, las propiedades donde se realizará los cambios de estilos.

Ejemplo:

```
h1{
    color:red;
}
```

SELECTORES DE CLASE

Se utiliza para crear un conjunto arbitrario de elementos (solamente que ellos a los que en el código html se le aplicará esta clase mediante el atributo class), aunque no compartan ser la misma etiqueta html (por ejemplo, el conjunto ". Título" se podría aplicar a un h1, a varios h2 y algunos h3), por un conjunto de "sólo algunos" ejemplares de una misma etiqueta (por ejemplo, el conjunto ". Destacado" aplicados solamente a tres o cuatro elementos h2, pero no a todos los h2).Para diferenciar de los otros selectores este se le antepone un punto y luego se indica el nombre de la clase.

Ejemplo de definición:

```
.importante{
        font-weight:bold;
        color:blue;
      }
```

SELECTORES DE CLASE

Ejemplo de llamado:

```
<div class=“importante”>texto</div>

<p class=“importante”>otra cosa de texto importante</p>
```

SELECTORES DE IDENTIFICADOR

Se utiliza para seleccionar un elemento que es único, que no puede repetirse dentro de una misma página, y que por eso mismo lo identifica. Cuando se define en una hoja de estilos se puede reconocer porque comienza por un numeral.

Ejemplo de definición:

```
#noticia1{
        float:left;
        }
```

Ejemplo de llamado:

```
<p id="noticia1">única cosa con este identificador</p>
```

SELECTORES MÚLTIPLES

Si se necesita definir exactamente las mismas reglas de estilos a varias cosas, es decir, estilos comunes a varios selectores diferentes. Consiste en declarar una o más reglas de estilos una sola vez entre las llaves, pero indicando que se apliquen a varios selectores a la vez, en una lista separada por comas previa a esas llaves. La coma indica el fin de un selector y el comienzo de otro. La idea es definir una sola vez aquello que tiene en común distintos selectores:

Ejemplo de definición:

```
h1, h2 , h3 , h4 {
        color:blue;
    }
```

Ejemplo de llamado:

```
<h1>texto de encabezado de color azul</h1>
```

SELECTORES MÚLTIPLES

Ejemplo de definición:

```
h1, .importante , #columna{
        border-botton:1px solid #000;
      }
```

Ejemplo de llamado:

```
<h1>texto de encabezado</h1>
<p class="importante">texto importante</p>
<p id="columna">única cosa con este identificador</p>
```

SELECTORES CONTEXTUALES

Estos electores relacionan dos selectores formando un nuevo selector único, más específico: de los electores implicados, el primero sirve de contexto, y el segundo es al que en realidad se apunta:

Ejemplo de definición:

```
Contexto elemento{
        propiedad1:valor_de_la_propiedad1;
        propiedad2:valor_de_la_propiedad2;
       }
```

Se lee de atrás para delante (los párrafos que estén dentro de #pie)

```
#pie p{
    propiedad1:valor_de_la_propiedad1;
    propiedad2:valor_de_la_propiedad2;
   }
```

```
<div id="pie">
    <p>primero que sí</p>
    <p>segundo que sí</p>
 </div>
<p>tercero que no</p>
```

SELECTORES DE ATRIBUTO

Existen otros tipos de selectores bastante más específicos, qué se basan en que el marcado del elemento apuntado tenga definido en el código html determinado atributo:

Ejemplo de definición:

```
elemento [atributo] {
                    }
```

Por ejemplo:

```
A[target]{
         }
```

SELECTORES DE VALOR DE ATRIBUTO

Cuando se desea seleccionar elementos del mercado que posean un mismo valor en alguno de sus atributos (por ejemplo: todos los enlaces que apuntan hacia el index.html, o todas las imágenes llamadas "logo.jpg" del sitio web), se puede utilizar un selector de valor de atributo, que sólo seleccionar aquellos elementos que en el atributo indicado posean exactamente el valor que se haya definido:

Ejemplo de definición:

```
elemento [atributo=valor] {
                          }
```

Por ejemplo:

```
A[href="index.html"]{
                    }
img[src="logo.jpg"]{                    }
```

SELECTORES DE VALOR DE ATRIBUTO

Ejemplo de definición:

```
elemento [atributo1="valor1"] [atributo2="valor2"] {
}
```

Por ejemplo:

```
A[href="index.html"]  [target="_blank"]{
}
```

SELECTORES DE VALOR DE ATRIBUTO

Ejemplo de definición:

```
elemento [atributo~=valor] {
}
```

Por ejemplo:

```
p[class~=otro]{
}
```

Ejemplo de llamado:

```
<p class="uno otro">texto</p>
<p class="otro tercero">texto</p>
```

SELECTORES COMBINADORES

Estas son las tres alternativas posibles:

- **Selectores de descendientes (A B)**
- **Selectores de hijos (A>B)**
- **Selectores de hermanos adyacentes (A+B)**

SELECTORES CONTEXTUALES

Estos electores relacionan dos selectores formando un nuevo selector único, más específico: de los electores implicados, el primero sirve de contexto, y el segundo es al que en realidad se apunta:

Ejemplo de definición:

```
Contexto elemento{
        propiedad1:valor_de_la_propiedad1;
        propiedad1:valor_de_la_propiedad1;
      }
```

Ejemplo de llamado:

```
<h1>texto de encabezado</h1>
<p class="importante">texto importante</p>
<p id="columna">única cosa con este identificador</p>
```