UD3 HTML Y CSS.

DISEÑO DE INTERFACES WEB

3.2 Nuevos elementos semánticos HTML5. Nuevas propiedades CSS3

Contenidos

1. Objetivos

En este segundo tema se pretenden conseguir los siguientes objetivos:

- Entender la **importancia de la semántica** de los elemento HTML5.
- Conocer las **nuevas etiquetas** de HTML5.
- Utilizar correctamente las nuevas **etiquetas estructurales** en función de su semántica.
- Conocer los **nuevos selectores** de CSS3.
- **Crear diseños web vistosos** utilizando degradados, sombras, transparencias, bordes redondeados y/o bordes con imágenes.

2. Introducción

En el tema 1 vimos una introducción a HTML5 y CSS3 haciendo especial hincapié en aquellas **cosas que no han cambiado** con respecto a las versiones anteriores de estas tecnologías.

En este tema veremos qué **nuevos elementos** se han introducido y por qué, así como otros elementos que ya existían cuya **semántica ha sido redefinida**.

En lo que respecta a CSS, veremos los nuevos **selectores** CSS3, **pseudo-elementos, pseudo-clases**, etc.

3. HTML5

HTML5 incorpora nuevos elementos que ayudan a **identificar** cada sección del documento y **organizar** el cuerpo del mismo. Las secciones más importantes son diferenciadas y la estructura principal ya no depende más de los elementos ** o <table>**.

Cómo usamos estos nuevos elementos depende de nosotros, pero las palabras clave otorgadas a cada uno de ellos nos **ayudan a entender** sus funciones.

Normalmente, una página o aplicación web está **dividida entre varias áreas** visuales para mejorar la experiencia del usuario y facilitar la interactividad. Las palabras claves que **representan** cada nuevo elemento de HTML5 están íntimamente relacionadas con estas áreas.

3. HTML5

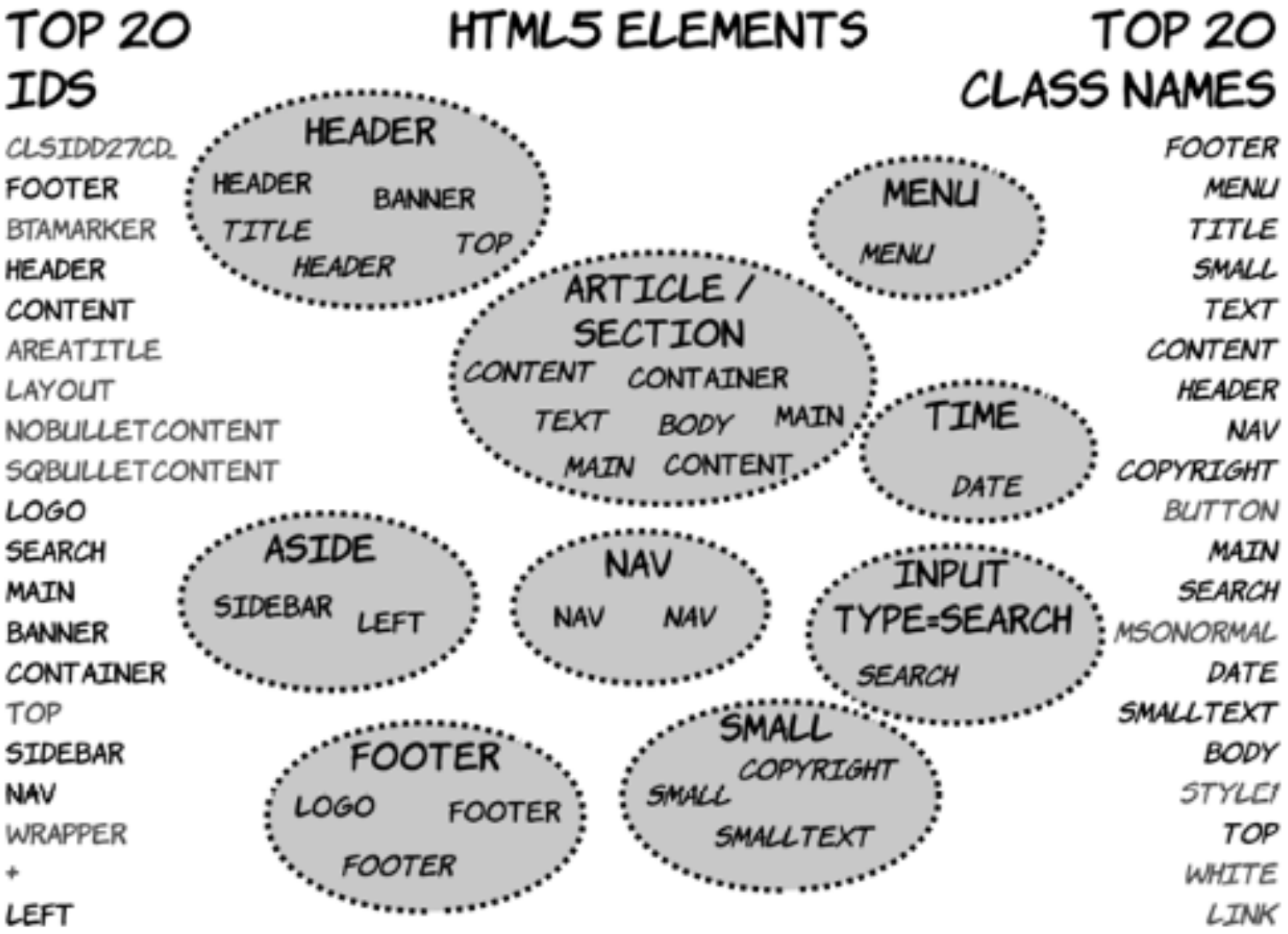
La semántica de los elementos

Como ya sabemos, HTML dispone de dos atributos que le permiten ampliar la semántica de los elementos: **id y class**.

El atributo id es un identificador **único**, y suele ser una palabra significativa con un **valor semántico**. Por el contrario, el atributo **class no es único**, y además, un elemento puede tener aplicadas **múltiples clases**.

En 2005 se realizaron varios estudios que analizaban cómo los autores estaban **usando los valores id y class** en la web. El diagrama que sigue muestra los **20 resultados que más veces aparecían** en cada categoría y los nuevos elementos de HTML5 correspondientes.

3. HTML5



Nuevos elementos HTML5 basados en los atributos class e id más utilizados

3. HTML5

Nuevos elementos estructurales

Como ya hemos visto, la mayoría de los sitios web tienen un diseño similar. La estructura suele dividirse en diferentes secciones donde suelen aparecer una cabecera, una barra de navegación, una zona de contenidos, etc.

1. Cabecera
2. Barra de Navegación
3. Sección de Información Principal
4. Barra Lateral
5. El pie o la barra Institucional

3. HTML5

Elemento header

Todo el contenido que acostumbrábamos a incluir dentro de un elemento , ahora se incluiría en un <header>.

La principal diferencia es que mientras que sólo podemos tener un elemento en toda la página, **no existe esta restricción** para el elemento <header>, se puede incluir un nuevo elemento de encabezado para introducir cada sección del sitio. Podemos interpretar una sección como cualquier parte de contenido que pueda necesitar su propio encabezado.

3. HTML5

Elemento `section`

El contenido del elemento de section debe ser "**temático**", por lo que sería incorrecto utilizarlo de una manera genérica para envolver piezas sin relación de contenido.

Algunos ejemplos de usos aceptables son:

- Secciones individuales de una interfaz con **pestañas**.
- Segmentos de una página "**Acerca de**", por ejemplo, la página "Acerca de" de una empresa podría incluir secciones sobre la historia de la empresa, su misión y su equipo.
- Diferentes partes de los "**términos de servicio**" de la página.
- Distintas secciones de un sitio de **noticias** en línea, por ejemplo, los artículos se podrían agrupar en secciones que cubren los **deportes**, los asuntos **mundiales**, y las noticias **económicas**.

3. HTML5

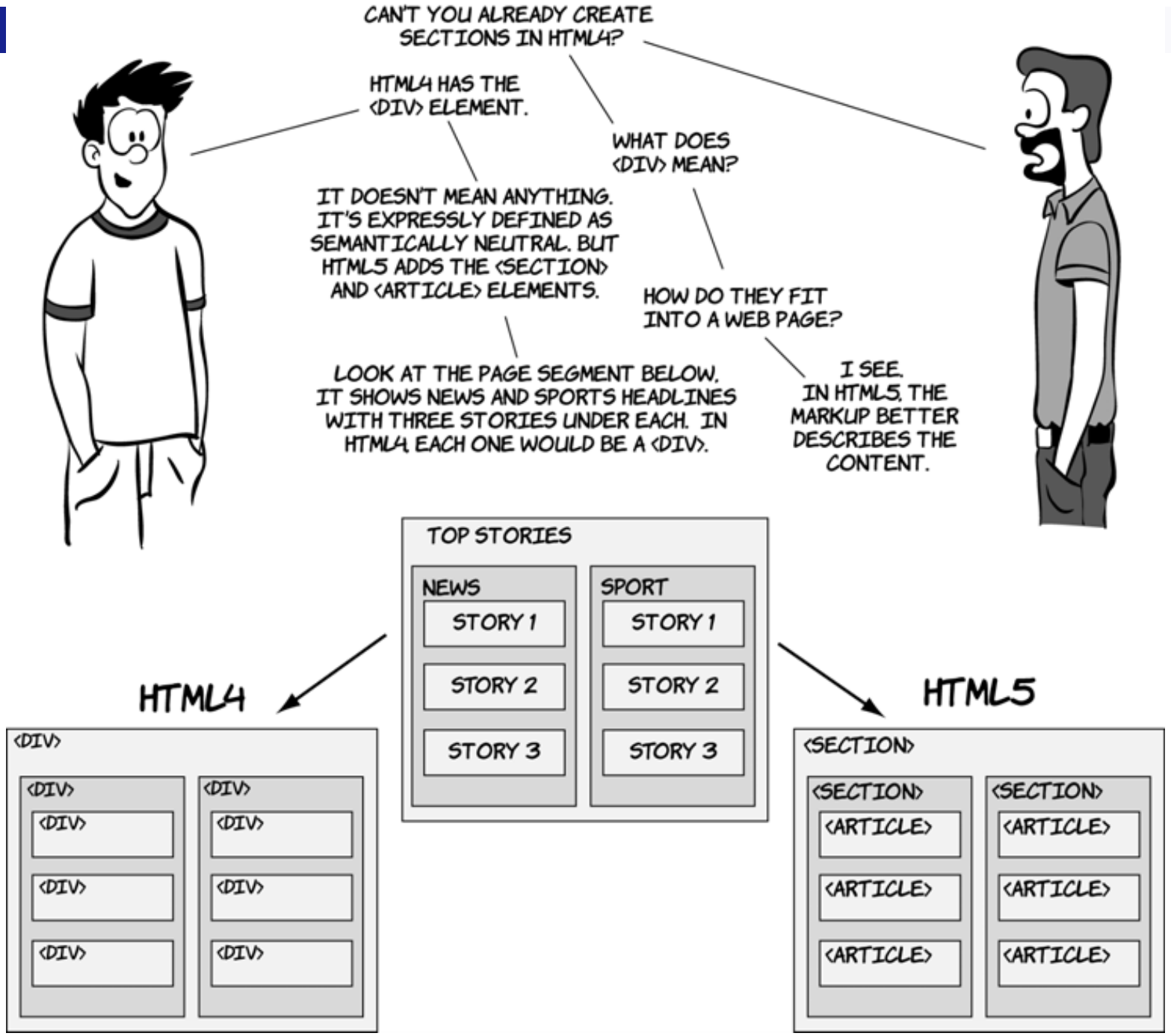
Elemento `article`

Similar a <section>, pero hay algunas diferencias notables. Mientras que <section> puede contener cualquier contenido que se puede agrupar temáticamente, <article> debe ser **una sola pieza de contenido** que puede valerse por sí misma.

Aquí hay algunas sugerencias de uso de <article>:

- Mensajes en el foro.
- Artículos de revistas o periódicos.
- Las entradas de un blog.
- Los comentarios enviados por los usuarios, etc.

Al igual que los elementos <section>, los elementos <article> **se pueden anidar**. También podemos anidar una sección dentro de un artículo, y viceversa.



Comparativa del uso de los nuevos elementos <section> y <article> con respecto a HTML4

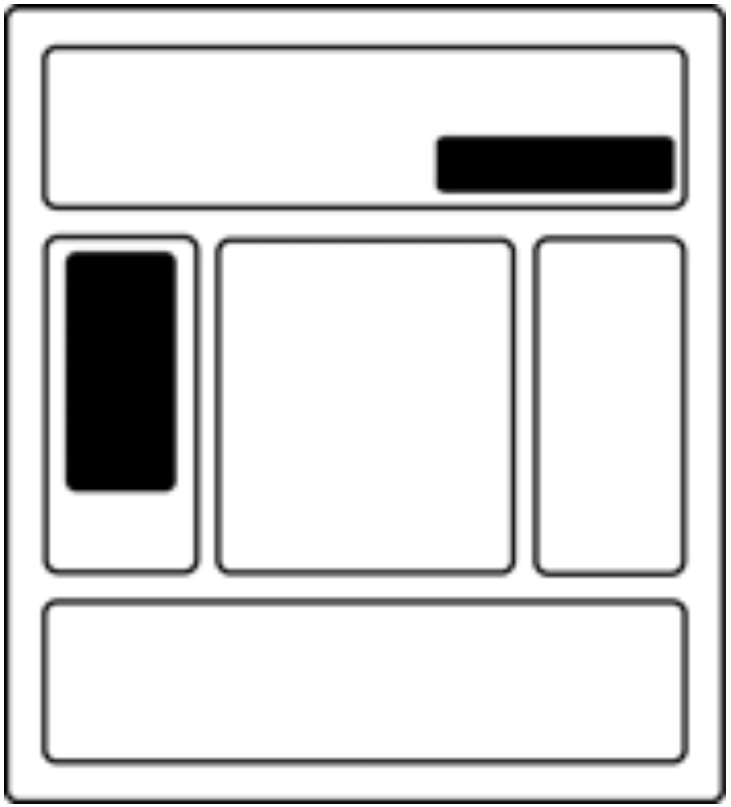
3. HTML5

Elemento nav

Podemos asumir que este elemento aparecerá en, prácticamente, todos los proyectos. <nav> representa un **grupo de vínculos de navegación**.

Lo más habitual será que contenga una **lista desordenada de enlaces**, aunque hay otras opciones.

En cualquier caso, el <nav> se debe reservar para la navegación que es de primordial importancia.

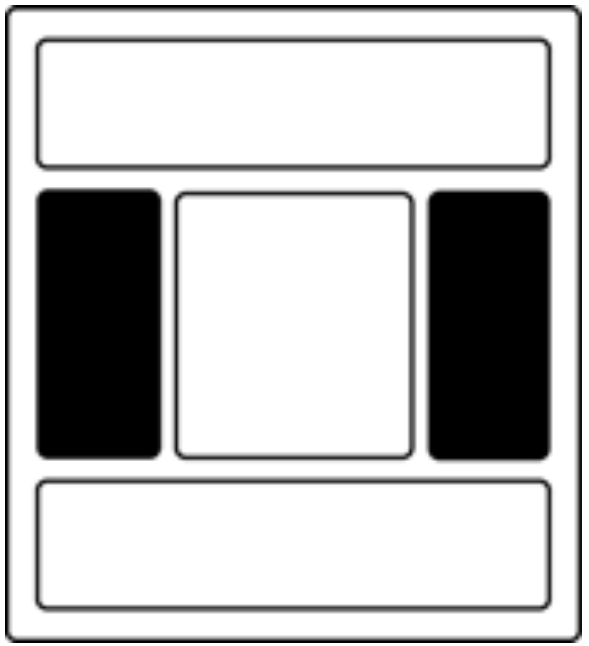


3. HTML5

Elemento `aside`

Este elemento representa una parte de la página que está "tangencialmente relacionado con el contenido que se encuentra alrededor, y que podría considerarse separado de ese contenido". Algunos usos posibles para un <aside> serían: **una barra lateral, una lista secundaria de enlaces, o un espacio para la publicidad**.

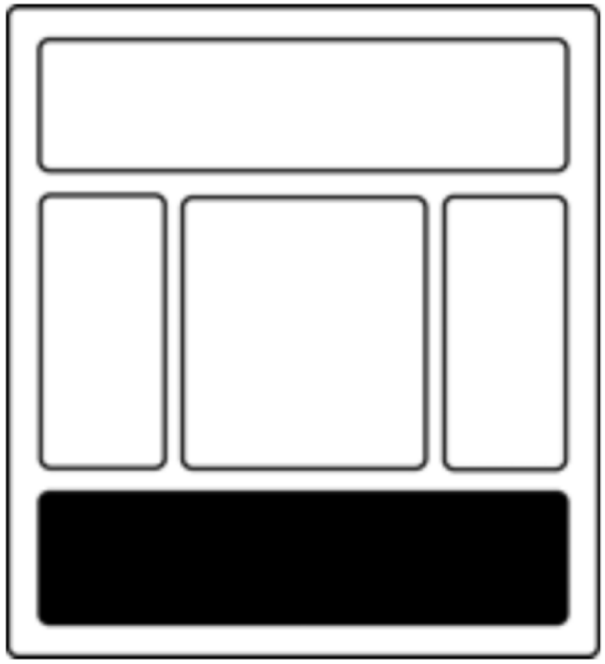
El elemento <aside> (como en el caso del <header>) **no se define por su posición en la página**. Podría ser en un "lado", o podría estar en otro lugar. Es el contenido en sí, y su relación con otros elementos, lo que lo define.



3. HTML5

Elemento `footer`

Al igual que con el <header>, **podemos tener varios** <footer> en una sola pagina. Representa un **pie de página de la sección de contenido** que es su ancestro más cercano. La "sección" de contenido podría ser todo el documento, un <section>, un <article> o un <aside>.



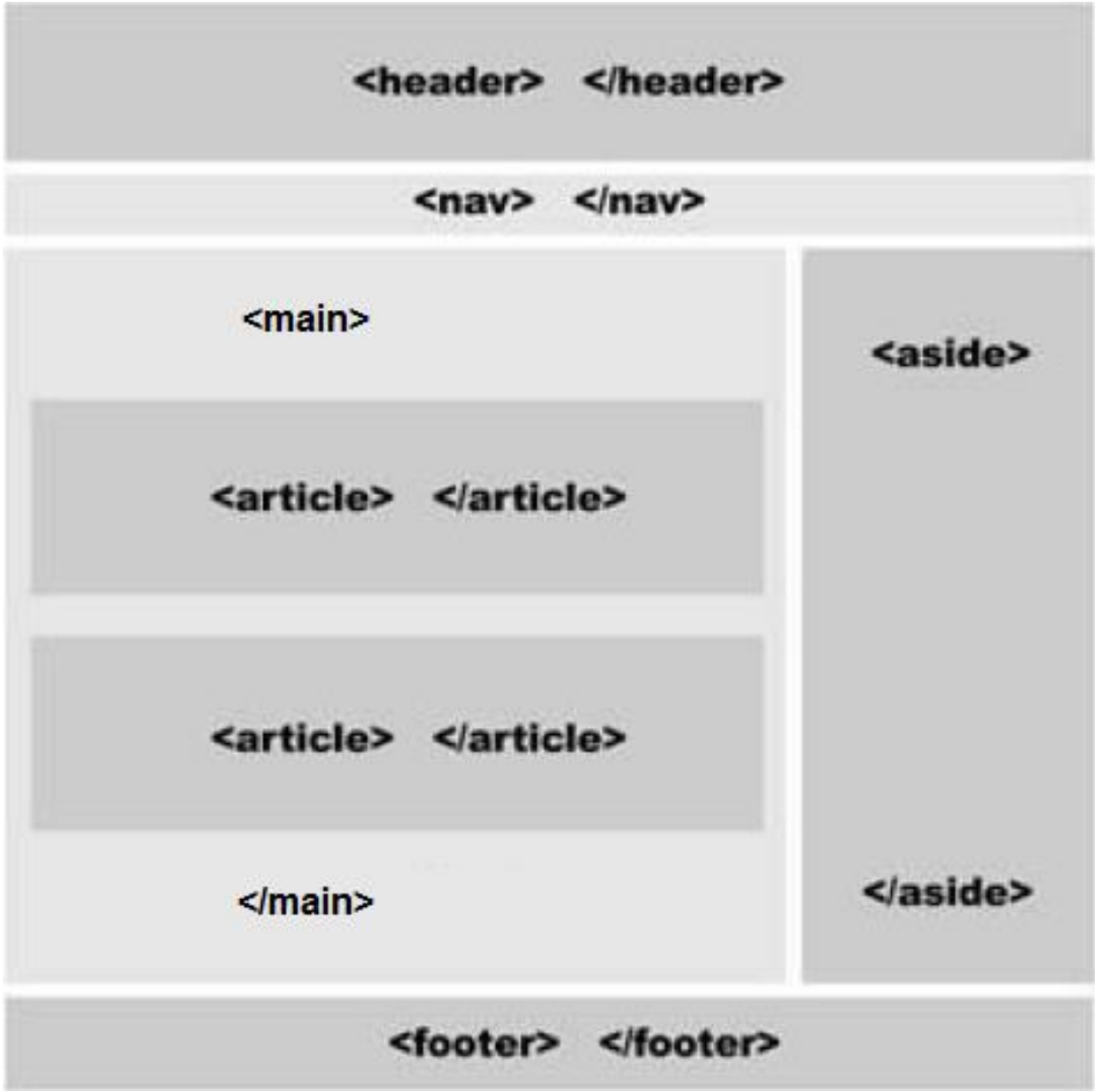
3. HTML5

Elemento `main`

El elemento <main> especifica el **contenido principal** del documento. Sus características principales son:

- Representa el **contenido principal del cuerpo** (<body>) de una web o aplicación.
- Incluye el **contenido que es único** en la página, y excluye contenido que se repite en todas las páginas de la web (menú, pie de página, barra lateral, etc.).
- **No** debe incluirse **más de un** elemento <main> por página.
- **No** debe incluirse el elemento <main> **dentro de** elementos como <article>, <aside>, <footer>, <header> o <nav>.

3. HTML5



Diseño típico utilizando elementos HTML5

3. HTML5

Otros elementos semánticos

Elementos figure y figcaption

La etiqueta `<figure>` especifica **contenido autónomo**, como ilustraciones, diagramas, fotos, listas de códigos, etc.

Si bien el contenido del elemento <figure> está relacionado con el flujo principal, su **posición es independiente** del flujo principal y, **si se elimina, no debería afectar** el flujo del documento.

El elemento `<figcaption>` se usa para **agregar un título** para el elemento <figure>.

```
<figure>
   <figcaption>Screen Reader Support for WAI-ARIA</figcaption>
   <img src="wai-aria.png" alt="WAI-ARIA">
</figure>
```

3. HTML5

Elemento mark

El elemento <mark> indica una **parte** del documento que ha sido **remarcada por la importancia** que tiene para la actividad actual del usuario.

Es cierto que podemos imaginar muy pocos usos para este elemento, posiblemente, el más habitual sea en el contexto de una **búsqueda**, donde queremos **destacar las palabras clave** que se han buscado dentro de los resultados.

```
<h1>Sí, usted puede utilizar <mark> HTML5 </mark> Hoy!</h1>
```

Sí, usted puede utilizar HTML5 Hoy!

3. HTML5

Elemento time

El elemento <time> ha sido diseñado específicamente para tratar con el problema de la lectura de las fechas y horas. Veamos el siguiente ejemplo:

```
<p>La conferencia de HTML5 será el próximo 12 de Octubre.</p>
```

Aunque cuando una persona lee el párrafo anterior tiene claro cuándo se va a producir el evento, si es una máquina la que está analizando la información, probablemente, encuentre problemas para deducirlo. A continuación, veremos el mismo párrafo, introduciendo el elemento <time>:

```
<p>La conferencia de HTML5 será el próximo <time datetime="2020-10-12">12 de Octubre</time>.</p>
```

3. HTML5

Elemento time

Veamos algunos ejemplos:

- Sin el atributo datetime el contenido debe ser valido:

```
<time>2009-11-13</time>
```

- Con el atributo datetime el contenido puede ser cualquier cosa:

```
<time datetime="2009-11-13">13 Noviembre</time>
<time datetime="20:00">8pm</time>
```

- Utilizando la zona horaria:

```
<time datetime="2009-11-13T20:00+00:00">8pm</time>
```

- utilizando la zona horaria “Z” (La zona “Z” la utilizamos para representar Universal Coordinated Time (UTC)):

```
<time datetime="2009-11-13T20:00Z">8pm</time>
```

3. HTML5

Elementos con la semántica redefinida

El elemento small

Previamente, este elemento era utilizado con la intención de presentar cualquier **texto con letra pequeña**. Esto hizo que dicho elemento **dejara de usarse**, ya que, con la aparición de las **páginas CSS** todo lo que estuviera relacionado con la presentación de los contenidos debía indicarse mediante CSS.

En HTML5, el nuevo propósito del elemento <small> es presentar la llamada letra pequeña como **impresiones legales, descargos, etc...**

```
<small>Derechos Reservados &copy;</small>
```

3. HTML5

Elementos con la semántica redefinida

El elemento cite

Otro elemento que ha cambiado su naturaleza para volverse más específico es <cite>. En versiones previas, se solía utilizar este elemento para **citar partes** de una obra. Ahora las etiquetas <cite> **encierran el título** de un trabajo, como un libro, una película, una canción, etc...

```
<span>Me encanta la película <cite>Gladiator</cite></span>
```

3. HTML5

Elementos con la semántica redefinida

El elemento address

El contenido de un elemento address provee **información de contacto** para el autor de la sección o documento.

El elemento address **NO debe ser utilizado para representar cualquier información de contacto**. Sólo la información relevante de contacto acerca del **autor** del documento o **sección** en la que se encuentre.

```
<article>
   <header>
      <h1>Título del mensaje </h1>
   </header>
   <p>Este es el texto del mensaje</p>
   <footer>
    <address>
         Escrito por: <a href="http://www.mario.com">Mario</a>
     </address>
  </footer>
</article>
```

4. CSS3

En este punto estudiaremos las **contribuciones hechas por CSS3 a HTML5**, y algunas de las propiedades que simplifican la vida de diseñadores y programadores.

Una nota sobre prefijos de proveedores

Hoy en día, si queremos utilizar muchas de las nuevas características de CSS3, estamos **obligados a incluir un buen número de líneas** adicionales de código.

Por ejemplo, para transformar un elemento en Firefox, es necesario utilizar la propiedad `-moz-transform`; para hacer lo mismo en los navegadores basados en WebKit, como Safari y Google Chrome, se utiliza la propiedad `-webkit-transform`.

4. CSS3

Los prefijos para los navegadores más comunes:

- `-moz-` para Firefox.
- `-webkit-` para Safari y Chrome.
- `-o-` para Opera.
- `-khtml-` para Konqueror.
- `-ms-` para Internet Explorer.
- `-chrome-` específico para Google Chrome.

Puede parecer que esto elimina algunos de los beneficios de CSS3, pero esto se hace así porque las **especificaciones aún están por definir**, y las primeras implementaciones tienden a tener errores.

Cuando las especificaciones estén perfectamente definidas y las propiedades refinadas, **se eliminarán las propiedades con prefijo**, propias de navegadores individuales.

4. CSS3

Si quieres ver los navegadores que soportan una determinada propiedad CSS o un elemento HTML5, puedes consultar la página.

http://caniuse.com

4. CSS3

Selector general de elementos hermanos

Su sintaxis es **`elemento1 ~ elemento2`** y selecciona el elemento2 que es hermano de elemento1 y se encuentra detrás en el código HTML.

```
h1 + h2 { ... } /* selector adyacente */
h1 ~ h2 { ... } /* selector general de hermanos */
<h1>...</h1>
<h2>...</h2>
<p>...</p>
<div>
     <h2>...</h2>
</div>
<h2>...</h2>
```

(h1 + h2) solo selecciona el primer elemento <h2> de la página, ya que es el único que cumple que es hermano de <h1> y se encuentra justo detrás en el código HTML.

(h1 ~ h2) selecciona todos los elementos <h2> de la página salvo el segundo. Aunque el segundo <h2> se encuentra detrás de <h1> no son elementos hermanos, ya que no tienen el mismo elemento padre.

4. CSS3

Pseudo-elementos

Los pseudo-elementos de CSS 2.1 se mantienen en CSS 3, pero cambia su sintaxis y **ahora se utilizan :: en lugar de :** delante del nombre de cada pseudo-elemento:

- `::first-line`, selecciona la **primera línea** del texto de un elemento.
- `::first-letter`, selecciona la **primera letra** del texto de un elemento.
- `::before`, selecciona la **parte anterior** al contenido de un elemento para insertar nuevo contenido generado.
- `::after`, selecciona la **parte posterior** al contenido de un elemento para insertar nuevo contenido generado.

4. CSS3

Pseudo-elementos

CSS 3 añade, además, un nuevo pseudo-elemento:

::selection, selecciona el **texto que ha seleccionado un usuario con su ratón o teclado**. Sólo unas pocas propiedades css pueden ser aplicadas a este selector, en concreto: color, background, cursor y outline.

```
::selection {color:red;background:yellow;}
```

Para que este selector funcione en **Firefox, debemos añadir el prefijo –moz-**, por lo tanto, para tener la funcionalidad disponible en todos los navegadores, haremos lo siguiente:

```
::selection {color:red;background:yellow;}
::-moz-selection {color:red;background:yellow;}
```

4. CSS3

Pseudo-clases

```
<p>fuera</p>
<div>
  <p>Mi texto1</p>
  <p>Mi texto2</p>
  <p>Mi texto3</p>
  <p>Mi texto4</p>
</div>
```

El código anterior contiene cinco elementos <p>, uno que es hijo del documento y cuatro que son hermanos entre sí e hijos del mismo elemento .

Usando pseudo-clases podemos **referenciar un elemento específico** sin importar cuánto conocemos sobre sus atributos y el valor de los mismos:

```
p:nth-child(1){ background: #999999; }
```

4. CSS3

Pseudo-clases

Si quisiéramos restringir la selección **sólo a los hijos del **, haríamos lo siguiente:

```
div p:nth-child(1){ background: #999999; }
```

También se pueden utilizar las palabras clave even y odd como índice para seleccionar los hijos **pares o impares** respectivamente:

```
div p:nth-child(even){ background: #999999; }
```

En este caso, se seleccionarán los párrafos 2 y 4.

4. CSS3

Pseudo-clases para la selección de hijos y hermanos

Psudo-clase	descripción
:nth-child(n)	Selecciona el enésimo hijo de su padre
:nth-last-child(n)	Selecciona el enésimo hijo de su padre contando desde el último
:nth-of-type(n)	Selecciona el enésimo hermano de su tipo
:nth-last-of-type(n)	Selecciona el enésimo hermano de su tipo comenzando desde el último
:first-child	Selecciona el primer hijo de su padre
:last-child	Selecciona el último hijo de su padre
:first-of-type	Selecciona el primer hermano de su tipo
:last-of-type	Selecciona el último hermano de su tipo
:only-child	Selecciona los que sean hijos únicos
:only-of-type	Selecciona los que sean los únicos hermanos de su tipo
:empty	Selecciona los elementos que no tienen hijos. Si un elemento contiene sólo texto no se considera vacío.

4. CSS3

Pseudo-clases

Otras pseudo-clases nuevas que también son importantes:

`:enabled, :disabled` Selecciona elementos de interfaz de usuario (formularios) según estén **activado o desactivado**.

`:checked` Selecciona elementos de (radio botones o casillas de verificación) que están en estado **checked**

La pseudo-clase :not() se utiliza realizar una negación:

```
:not(p) { margin: 0px; }
```

Este selector asignará un margen de 0 píxeles a todos los elementos del documento **excepto** los elementos <p>. Podemos utilizar cualquier selector válido:

```
:not(.mitexto) -> Todos menos los de la clase mitexto
:not(#menu) → Todos excepto el id menu
```

4. CSS3

Nuevas propiedades CSS3

Para facilitar el aprendizaje de las nuevas propiedades CSS3 vamos a ir aplicándolas a documento de ejemplo.

Descarga del aula virtual el documento **ejemplo.html** y **estiloscss3.css** sobre los que haremos una serie de ejercicios.

Dentro del cuerpo del HTML, sustituye **PCX** por el número de tu PC. Al final del tema, se te solicitará que entregues un **documento con capturas** de pantalla de cada ejercicio realizado.

4. CSS3

estiloscss3.css

No hay nada nuevo en estas reglas. Vamos a destacar el uso de la propiedad **display: block** para asegurarnos de que el navegador interpreta el elemento <header> como un elemento en bloque. Para ver más usos de esta propiedad se recomienda consultar la siguiente URL: http://librosweb.es/css_avanzado/capitulo_4/propiedad_display.html

Si visualizamos la página se verá así:

4. CSS3

Bodes con esquinas redondeados (border-radius)

Gracias a la propiedad border-radius de CSS3, ahora podemos hacerlo de forma muy sencilla (si queremos asegurarnos la **compatibilidad con el mayor número de navegadores** posible, debemos usar los prefijos `-moz-` y `-webkit-`).

Si **todas las esquinas tienen la misma curvatura** podemos utilizar un solo valor. Las siguientes propiedades las añadiremos a la regla #principal de nuestro ejemplo:

```
-moz-border-radius: 20px;
-webkit-border-radius: 20px;
border-radius: 20px;
```

Curso HTML5 y CSS3

4. CSS3

Bodes con esquinas redondeados (border-radius)

Sin embargo, como ocurre con las propiedades `margin` y `padding`, podemos también declarar un **valor diferente para cada esquina**.

Los valores se aplicarán en el siguiente orden: esquina superior izquierda, esquina superior derecha, esquina inferior derecha y esquina inferior izquierda:

```
-moz-border-radius: 20px 10px 30px 50px;
-webkit-border-radius: 20px 10px 30px 50px;
border-radius: 20px 10px 30px 50px;
```

Curso HTML5 y CSS3

4. CSS3

Bodes con esquinas redondeados (border-radius)

También podemos dar forma a las esquinas declarando un **segundo grupo de valores** separados por una barra. Los valores a la izquierda de la barra representarán el radio horizontal, mientras que los valores a la derecha representan el radio vertical. La combinación de estos valores genera una elipse:

```
-moz-border-radius: 50px / 20px;
-webkit-border-radius: 50px / 20px;
border-radius: 50px / 20px;
```

Curso HTML5 y CSS3

Para probar de una forma sencilla cómo quedaría esta propiedad puedes utilizar la herramienta disponible en la url: http://www.cssmatic.com/border-radius

4. CSS3

Sombras en las cajas (box-shadow)

Gracias a la nueva propiedad box-shadow podemos aplicar sombras a nuestras cajas con una simple línea de código.

La propiedad box-shadow necesita al menos **tres valores**: Los dos primeros valores, expresados en píxeles, establecen el **desplazamiento horizontal y vertical** (positivo o negativo) de la sombra y el tercero indicará el **color** de la misma.

```
-moz-box-shadow: 5px 5px rgb(150,150,150);
-webkit-box-shadow: 5px 5px rgb(150,150,150);
box-shadow: 5px 5px rgb(150,150,150);
```

4. CSS3

Sombras en las cajas (box-shadow)

Otro valor que se puede agregar a la propiedad antes de indicar el color es la **distancia de difuminación**. Este será un valor expresado en píxeles que indicará como se va a difuminar la sombra:

```
-moz-box-shadow: 5px 5px 10px rgb(150,150,150);
-webkit-box-shadow: 5px 5px 10px rgb(150,150,150);
box-shadow: 5px 5px 10px rgb(150,150,150);
```

Curso HTML5 y CSS3

4. CSS3

Sombras en las cajas (box-shadow)

El último valor posible para box-shadow no es un número, sino una **palabra clave**: `inset`. Esta palabra clave la indicaremos después del color, y convierte la sombra externa en una **sombra interna**, lo cual provee un efecto de profundidad al elemento afectado.

```
-moz-box-shadow: 5px 5px 10px rgb(150,150,150) inset;
-webkit-box-shadow: 5px 5px 10px rgb(150,150,150) inset;
box-shadow: 5px 5px 10px rgb(150,150,150) inset;
```

Curso HTML5 y CSS3

4. CSS3

Sombras en los textos (text-shadow)

Los valores para text-shadow son similares a los usados para box-shadow. Podemos declarar la **distancia horizontal y vertical** de la sombra con respecto al objeto, el radio de **difuminación** y el **color** de la sombra.

```
#titulo {
   font: bold 36px verdana, sans-serif;
   text-shadow: 5px 5px 5px rgb(150,150,150);
}
```

Para probar de una forma sencilla cómo quedaría esta propiedad puedes utilizar la herramienta disponible en la url: http://css3gen.com/text-shadow/

4. CSS3

Fondos con gradiente

Los gradientes son uno de los efectos más utilizados en la web. **Anteriormente**, era necesario el uso de **imágenes** para lograr este efecto, pero ahora es realmente **fácil de hacer usando CSS**.

Los gradientes son configurados como **fondos**, por lo que podemos usar las propiedades `background` o `background-image` para declararlos.

Tenemos dos tipos de gradientes:

- gradiente lineal
- gradiente radial

4. CSS3

Gradiente lineal

Los gradientes lineales los indicamos con la **función linear-gradient**, que recibe los siguientes parámetros:

- Punto de comienzo: indica el punto donde comenzará el gradiente. Puede ser especificado en **píxeles**, **porcentaje** o usando las palabras clave **top, bottom, left y right**. También puede ser reemplazado por un **ángulo** para declarar una dirección específica del gradiente.
- Color inicial: indica con que color comenzará el gradiente.
  Color final: indica con que color terminará el gradiente.

Con el parámetro punto de comienzo, debemos tener en cuenta que hay una pequeña **diferencia entre la función estándar y las especificas** de los navegadores: en la **función estándar** debemos indicar la **palabra to y la dirección** del gradiente, mientras que en las **específicas** de los navegadores indicamos la **posición de comienzo** del gradiente.

4. CSS3

Gradiente lineal

```
background: -webkit-linear-gradient(top, #FFFFFF, #006699);
background: -moz-linear-gradient(top, #FFFFFF, #006699);
background: linear-gradient(to bottom, #FFFFFF, #006699);
```


```
background: -webkit-linear-gradient(30deg, #FFFFFF, #006699);
background: -moz-linear-gradient(30deg, #FFFFFF, #006699);
background: linear-gradient(30deg, #FFFFFF, #006699);
```

4. CSS3

Gradiente lineal

También podemos declarar los puntos de terminación para cada color de la siguiente forma:

```
background: -webkit-linear-gradient(30deg, #FFFFFF 50%, #006699 90%);
background: -moz-linear-gradient(30deg, #FFFFFF 50%, #006699 90%);
background: linear-gradient(30deg, #FFFFFF 50%, #006699 90%);
```

4. CSS3

Gradiente radial

Sólo difiere en unos pocos aspectos con respecto a la anterior. Debemos usar la función radial-gradient() y un **nuevo atributo para la forma**:

```
background:-webkit-radial-gradient(circle, #FFFFFF 0%, #006699 200%);
background:-moz-radial-gradient(circle, #FFFFFF 0%, #006699 200%);
background:radial-gradient(circle, #FFFFFF 0%, #006699 200%);
```

Existen dos posibles valores para la forma: **circle y ellipse** (por defecto será ellipse). Además, para cada color se indica la **posición** donde las transiciones **comienzan** (en el ejemplo, empieza en la posición 0% y **terminan** en la 200%).

4. CSS3

Transparencias

Tenemos dos formas de definir las transparencias: **expresando el color** del elemento con transparencia o indicando un valor que indicará la **opacidad** del elemento.

RGBA

CSS3 ha agregado una nueva función llamada rgba() que simplifica la asignación de colores y transparencias.

La función rgba() tiene **cuatro atributos**. Los tres primeros son similares a los usados en rgb() y, simplemente, declaran los valores para los colores rojo, verde y azul en números decimales del 0 al 255. El **último**, en cambio, corresponde a la nueva **capacidad de opacidad**. Este valor se debe encontrar dentro de un **rango que va de 0 a 1**, con 0 como totalmente transparente y 1 como totalmente opaco.

4. CSS3

Transparencias

RGBA

```
#titulo {
    font: bold 36px verdana, sans-serif;
    text-shadow: 5px 5px 5px rgba(0,0,0,0.5);
}
```

Reemplazamos la función rgb() por rgba() en la sombra del título y agregamos un valor de **opacidad/transparencia de 0.5**. Ahora, la sombra de nuestro título se mezclará con el fondo creando un efecto mucho más natural.

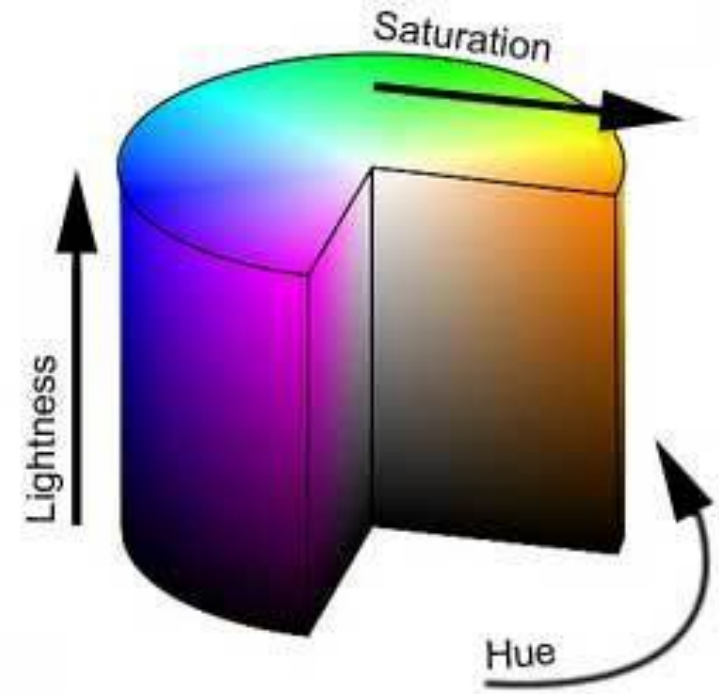
4. CSS3

Transparencias

HSLA

Del mismo modo que la función rgba() agrega un valor de opacidad a rgb(), la función hsla() lo agrega a hsl().

La función hsla() es, simplemente, otra función para generar colores que, en lugar de utilizar distintos valores para los colores rojo, verde y azul, **utiliza distintos valores de tono, saturación y luminosidad**.



4. CSS3

Transparencias

HSLA

La sintaxis es la siguiente: hsla(tono, saturación, luminosidad, opacidad).

- tono representa el color **extraído de un círculo** de color imaginario y es expresado en grados desde 0 a 360. **Cerca de 0 y 360 están los colores rojos, cerca de 120 los verdes y cerca de 240 los azules**.
- saturación es representado en **porcentaje**, desde 0% (escala de grises) a 100% (todo color o completamente saturado).
- luminosidad es también un valor en **porcentaje** desde 0% (completamente oscuro) a 100% (completamente iluminado). El valor 50% representa luminosidad normal o promedio.
- opacidad funciona igual que en la función rgba().

4. CSS3

Transparencias

HSLA

Si utilizamos esta función para cambiar el color de la fuente de nuestro título, obtendremos el siguiente resultado:

```
color: hsla(120, 100%, 50%, 0.5);
```

4. CSS3

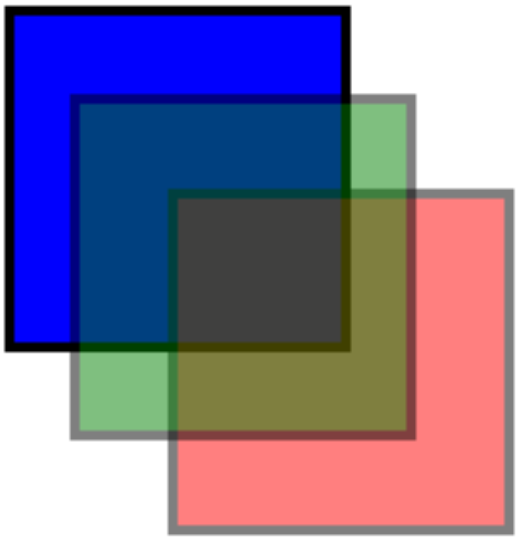
Transparencias

Propiedad opacity

El valor de la propiedad opacity se establece mediante un **número comprendido entre 0.0 y 1.0** (al igual que antes, 0.0 será transparente y 1.0 totalmente opaco).

En el siguiente ejemplo, se establece la propiedad opacity con un valor de 0.5 para conseguir una transparencia del 50% sobre dos de los elementos :

```
#segundo, #tercero { opacity: 0.5; }
#primero { background-color: blue; }
#segundo { background-color: red; }
#tercero { background-color: green; }
```



4. CSS3

Outline

Propiedad de CSS que ha sido expandida en CSS3 para incluir un valor de desplazamiento. Esta propiedad se usaba para **crear un segundo borde**, y ahora ese borde puede ser mostrado **alejado del borde real** del elemento.

Por ejemplo, podemos añadir un segundo borde de 2 píxeles y un desplazamiento de 15 píxeles a la caja de nuestro título, si ponemos las siguientes propiedades en la regla `#principal`:

```
outline: 2px dashed #000099;
outline-offset: 15px;
```

4. CSS3

Border-image

La nueva propiedad border-image, ofrece la posibilidad de utilizar **imágenes propias** como borde de imagen.

Imagen de borde: border-image-source

La propiedad border-image-source **establece la imagen a utilizar** como imagen de borde. El elemento tiene que tener **definido un borde**[1] y la imagen se amplía o reduce para mostrarse completa en función del ancho que le hayamos puesto a la propiedad border. Si sólo se utiliza esta propiedad, la imagen se muestra en las cuatro esquinas del elemento. Ver siguiente ejemplo:

[1]Es importante que tenga también definido un estilo de borde (propiedad border-style), ya que, si no, no se mostrará correctamente en Firefox

4. CSS3

Border-image

```
#principal {
	display: block;
	width: 500px;
	margin: 50px auto;
	padding: 15px;
	text-align: center;
	border: 50px;
	border-style:solid;
	border-image-source: url("fo_bola_100.png");
}
```

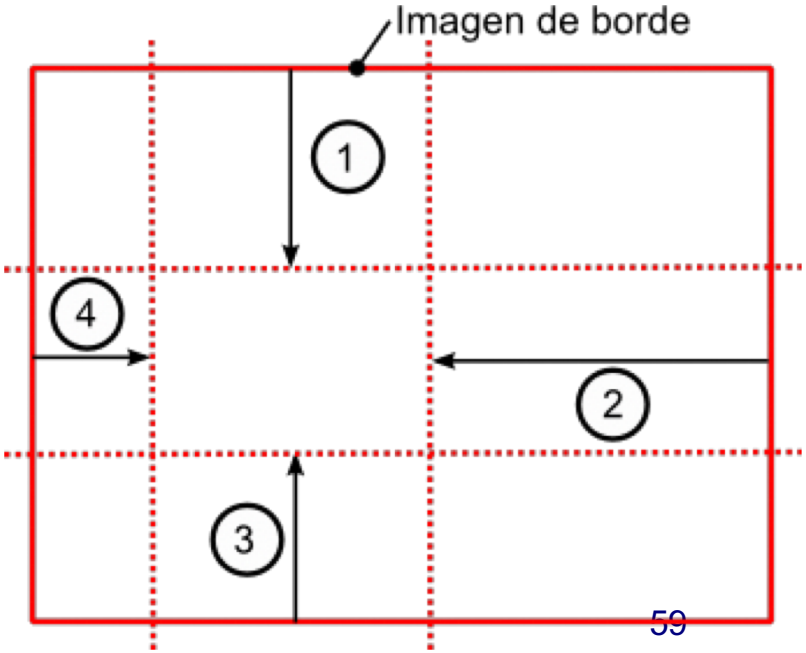
4. CSS3

Border-image

Troceado de la imagen de borde: border-image-slice

La propiedad border-image-slice permite trocear la imagen de borde, de manera que **cada trozo se coloque en un lado** del borde. Los posibles valores son de uno a cuatro valores en píxeles que indican a qué distancia de los bordes se recorta la imagen.

Cada uno de los ocho trozos (descartando el trozo central) se utiliza en cada una de las ocho zonas correspondientes del borde. Podemos indicar sólo dos valores, o incluso, un único valor



4. CSS3

Border-image

```
border: 27px;
border-style:solid;
border-image-source: url("fo_pastilla_60.png");
border-image-slice: 27;
```


```
border-width: 50px 80px 20px 80px;
border-style:solid;
border-image-source: url("fo_ventana.png");
border-image-slice: 50 80 20 80;
```

4. CSS3

Border-image

fill: este valor hace que la parte central de la imagen se utilice para rellenar el elemento.

```
border: 27px;
border-style:solid;
border-image-source: url("fo_pastilla_60.png");
border-image-slice: 27 fill;
```

4. CSS3

Border-image

Repetición de la imagen de borde: border-image-repeat

La propiedad border-image-repeat indica de qué forma se realizará el **relleno de los bordes** del elemento. Los posibles valores son:

stretch: la imagen **se estira o encoge** para ocupar todo el espacio necesario border: 40px;

```
border-style:solid;
border-image-source: url("fo_hormigas_210.png");
border-image-slice: 70;
border-image-repeat: stretch;
```

4. CSS3

Border-image

Repetición de la imagen de borde: border-image-repeat

repeat: la **imagen se repite** para ocupar todo el espacio necesario

round: la imagen se repite el máximo número **entero** de veces posible hasta rellenar el espacio, la **imagen será reescalada para que ninguna de las repeticiones quede cortada**.