

## HTML 5 y CSS 3. Nivel II

junio, 2018



## Objetivos del nivel

- Aprender a utilizar el modelo de caja
- Dar estilo al sitio web con CSS3
- Selectores avanzados CSS
- Posicionamiento

## Prerrequisitos del nivel

- HTML 5 y CSS 3 Nivel I

## Acerca de este manual

Este manual pertenece al Centro de Asesoramiento y Desarrollo Informático C.A. (CADI F1). Para obtener más información sobre este u otros cursos visite nuestra sitio Web [www.cadif1.com](http://www.cadif1.com), escribanos a la dirección de correo [cadi@cadif1.com](mailto:cadi@cadif1.com) o visítenos en nuestra sede ubicada en la Av. Pedro León Torres con calle 59, Centro Comercial Sotavento, piso 2 oficina 27, Barquisimeto estado Lara, Venezuela. Tlf. 0251-7179247, 0251-4410268.

Las marcas mencionadas en este manual son propiedad de sus respectivos dueños. Copyright 2018. Todos los derechos reservados.



## Contenido del nivel

### Capítulo 1. Modelo de Caja. Parte 1

- 1.1.- Modelo de Caja.
- 1.2.- Propiedad Border.
- 1.3.- Propiedad Height y Width.

### Capítulo 2. Modelo de Caja. Parte 2

- 2.1.- Propiedad Margin.
- 2.2.- Propiedad Padding.

### Capítulo 3. Posicionamiento de Elementos

- 3.1.- Posicionamiento y Estilo a Las Imágenes.
- 3.2.- Elementos Flotantes.
- 3.3.- Limpiar Los Elementos Flotantes.

### Capítulo 4. Posicionamiento Exacto de Elementos

- 4.1.- Posición absoluta (absolute).
- 4.2.- Posición Fija (fixed).

### Capítulo 5. Posicionamiento de Elemento Según su Flujo

- 5.1.- Introducción.
- 5.2.- Posición Normal (static).
- 5.3.- Posición Relativa (relative).

### Capítulo 6. Control del orden de apilamiento y comportamiento

- 6.1.- Apilamiento de Cajas (z-index).
- 6.2.- Contenido Excedente (overflow).

## Capítulo 7. Visualización

- 7.1.- Tipo de Caja (display).
- 7.2.- Visibilidad Del Elemento (visibility).

## Capítulo 8. Selectores Avanzados. Parte 1

- 8.1.- Comunes (clase, id y tipo).
- 8.2.- Hijos Descendientes.
- 8.3.- Hijos directos.

## Capítulo 9. Selectores Avanzados. Parte 2

- 9.1.- Introducción.
- 9.2.- Hermanos.
- 9.3.- Hermanos adyacentes.

## Capítulo 10. Selectores Avanzados Según el Atributo

- 10.1.- Introducción.
- 10.2.- Atributo presente.
- 10.3.- Atributo igual.
- 10.4.- Atributo contenido.

## Capítulo 11. Selectores Con Pseudo-clases

- 11.1.- Pseudo-clases.
- 11.2.- Estructura y posición.

## Capítulo 12. Efectos Especiales. Parte 1

- 12.1.- Border-radius.
- 12.2.- Box-shadow.
- 12.3.- Text-shadow.

## Capítulo 13. Efectos Especiales. Parte 2

13.1.- Multiple background.

## Capítulo 14. Transformación de Elemento

14.1.- Transformaciones.

## Capítulo 15. Transiciones de Elementos

15.1.- Transiciones.

15.2.- Prefijos Css en Los Navegadores.



## Capítulo 1. MODELO DE CAJA. PARTE 1

### 1.1.- Modelo de Caja

Uno de los principios necesarios para entender completamente HTML y CSS es el modelo de caja. El concepto de modelo de caja indica que todos los elementos de una página es una caja rectangular, y puede incluir los márgenes, rellenos y bordes.

Tener una comprensión general del modelo de caja es crucial ya que los sitios web están construidos principalmente en torno a ella. Obtener una comprensión del modelo de caja puede ser frustrante y difícil, pero necesaria para la construcción de sitios web. Además, saber cómo colocar los elementos en una página para crear un diseño es igualmente importante. Hay muchas maneras diferentes de colocar los elementos, cada uno de ellos depende de los contenidos y las circunstancias.



Ya sabemos, cada elemento es una caja rectangular, de las cuales incluye una altura y anchura, y puede tener diferentes márgenes, rellenos y bordes. Todos estos valores juntos construyen lo que se conoce como el modelo de caja.

La caja comienza con la altura y el ancho de un elemento, que puede ser determinado por el tipo de elemento, el contenido del elemento, o por las propiedades de ancho y altura especificada. La altura y la anchura es seguido por el relleno y el borde.

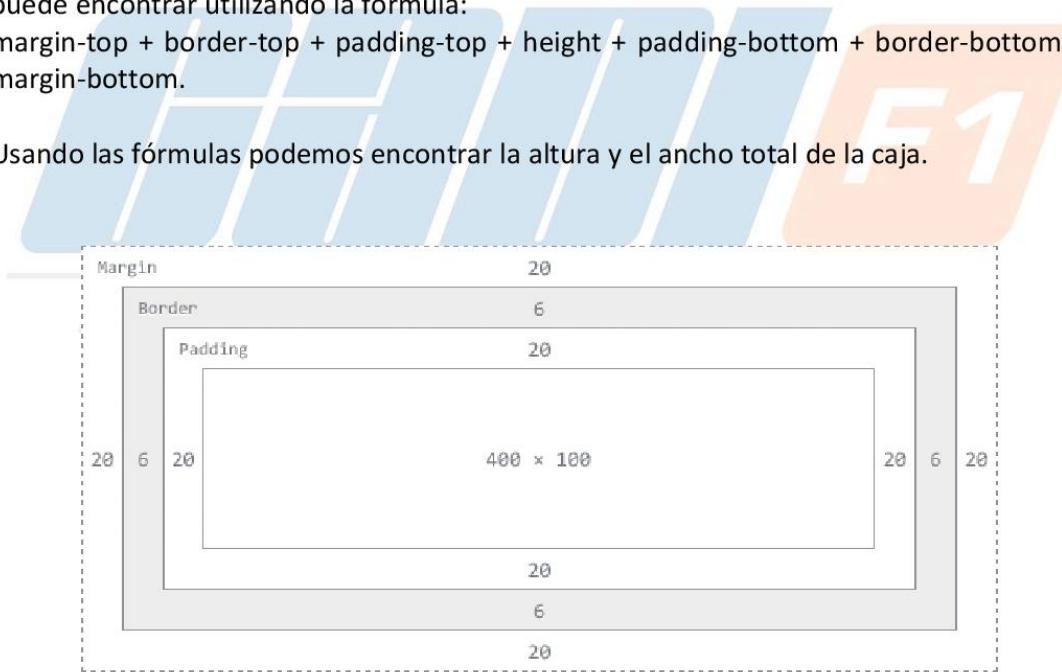
- Para sacar la anchura total de un elemento, incluyendo el modelo de caja, utilice la siguiente fórmula:

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

- En comparación, la altura total de un elemento, incluyendo el modelo de caja, se puede encontrar utilizando la fórmula:

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

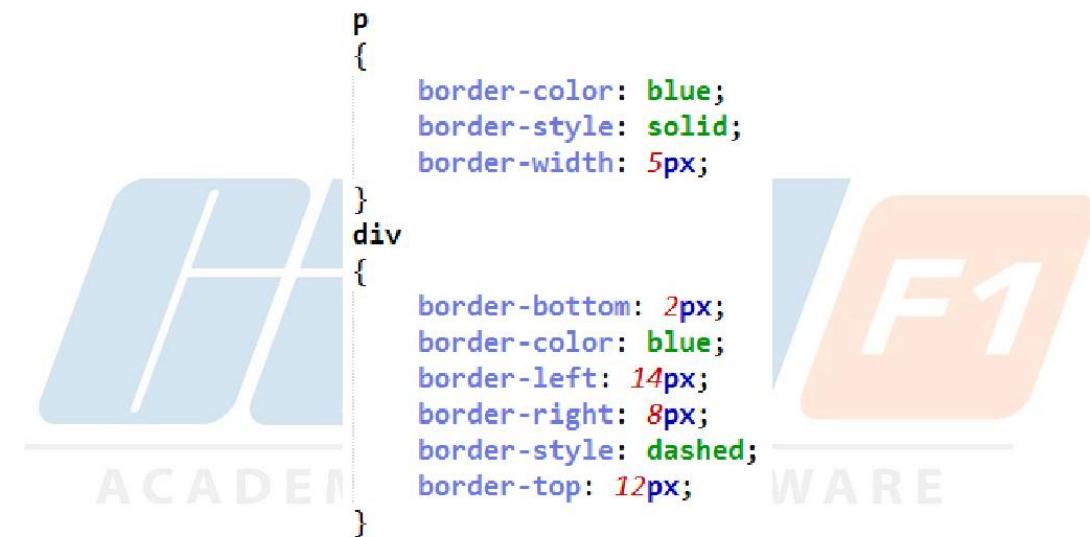
Usando las fórmulas podemos encontrar la altura y el ancho total de la caja.



## 1.2.- Propiedad Border

La propiedad border se sitúa entre el relleno y el margen y proporcionan un contorno alrededor de un elemento. Cada borde tiene tres valores, un ancho, estilo y color: border-width, border-style y border-color.

Por lo general, verá un ancho, un tipo de estilo y un color. Sin embargo tienen la capacidad de colocar en varios anchos, estilos y colores.



## 1.3.- Propiedad Height y Width

Cada elemento tiene una altura y anchura heredada. Dependiendo de la finalidad del elemento, la altura y la anchura puede ser predeterminada.

Si un elemento es clave para la distribución y el diseño de una página, puede requerir una altura y ancho específico. En este caso, los valores predeterminados para los elementos de bloque pueden ser anulados.

La altura (Height) por defecto de un elemento se determina por su contenido. Un elemento se expande y se contraen verticalmente según sea necesario para adaptarse a su contenido.

Para establecer una altura específica para un elemento de nivel de bloque se utiliza la propiedad height.

```
div
{
    height: 100px;
}
```

El ancho (Width) predeterminado de un elemento depende de la pantalla. Elementos a nivel de bloque disponen de un ancho predeterminado de 100%, el consumo de todo el espacio horizontal disponible.

Los elementos en línea se expanden y contraen horizontalmente para ocupar su contenido.

Dado que los elementos de nivel de línea no pueden tener un tamaño fijo, la propiedad width, al igual que con la propiedad height, sólo es relevante para colocar un ancho a un elemento.

```
div
{
    width: 100px;
}
```

## Capítulo 2. MODELO DE CAJA. PARTE 2

### 2.1.- Propiedad Margin

La propiedad margin nos permite establecer la longitud del espacio que rodea a un elemento. Los márgenes están fuera de cualquier elemento y son completamente transparentes.

Los márgenes pueden ser usados para ayudar a colocar los elementos en un lugar determinado en una página o simplemente para proporcionar espacio para respirar, manteniendo todos los demás elementos a una distancia segura.

### 2.2.- Propiedad Padding

La propiedad de relleno (padding) es muy similar a la de la propiedad margin, sin embargo, recae dentro de cualquier elemento. Juntas también heredaran los fondos de un elemento.

El relleno se utiliza para proporcionar una separación dentro de un elemento, no para el posicionamiento de un elemento como la propiedad margin.

```
div
{
    padding: 20px;
}
```

## Capítulo 3. POSICIONAMIENTO DE ELEMENTOS

### 3.1.- Posicionamiento y Estilo a Las Imágenes

Las imágenes pueden ser colocadas en un buen número de maneras diferentes, incluyendo en línea, bloque, alineado a la izquierda, a la derecha.

Todas estas posiciones se obtienen utilizando la propiedad float de CSS, junto con otros del modelo de caja, incluyendo margin, padding, border, y display.



```
img
{
    background: lightcyan;
    border: 3px solid green;
    padding: 8px;
}
```

### 3.2.- Elementos Flotantes

El modelo de caja es sólo la mitad de la batalla para la codificación de un diseño de página. La otra mitad consiste en saber cómo alinear correctamente todos los diferentes elementos de la página. Una manera de colocar los elementos uno al lado de otro, es utilizar la propiedad float. La propiedad float permite tomar elementos y flotar a la derecha o a la izquierda.

```
img
{
    float: left;
}
p
{
    float: right;
}
```

Hay algunas cosas importantes a tener en cuenta cuando los elementos se flotan. Lo primero, al flotar un elemento se va a flotar todo hasta el borde de su contenedor principal. Además, al flotar un elemento lo demás elementos comenzarán a alinearse a su alrededor en el flujo natural de la página.



### 3.3.- Limpiar Los Elementos Flotantes

Cada vez que un elemento se flota, se rompe el flujo normal de una página y los otros elementos se ajustarán alrededor según sea necesario. A veces esto es bueno, como cuando flotando una imagen al lado de un bloque de contenido, y a veces esto es malo.

Para devolver el documento a su flujo normal se utiliza la propiedad clear. La propiedad clear limpia cada elemento flotante hasta ese momento.

```
h1
{
  clear: none;
  clear: left;
  clear: right;
  clear: both;
}
```

## Capítulo 4. POSICIONAMIENTO EXACTO DE ELEMENTOS

### 4.1.- Posición absoluta (absolute)

Los elementos posicionados con el valor absolute, también aceptan los valores de desplazamiento de la caja: top, bottom, left o right; sin embargo no siguen el flujo normal del documento.

The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab displays the following code:

```
1 <div class="absolute">Caja Nro. 1</div>
2 <div>Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>
```

The 'CSS' tab displays the following code:

```
1 div
2 {
3     height: 50px;
4     outline: 1px solid black;
5     width: 100px;
6 }
7 .absolute
8 {
9     left: 20px;
10    outline: 1px solid red;
11    position: absolute;
12    top: 20px;
13 }
```

Below the code, the browser window shows the visual output. It consists of four rectangular boxes labeled 'Caja Nro. 1' through 'Caja Nro. 4'. 'Caja Nro. 1' is positioned at the top-left, while 'Caja Nro. 2', 'Caja Nro. 3', and 'Caja Nro. 4' are stacked vertically below it. A red box highlights 'Caja Nro. 1', and a red arrow points from the word 'A' on the left towards it, indicating its absolute position relative to the page.

Al sacar el elemento del flujo normal, los elementos se posicionan directamente en relación con su padre, el cual debe tener la posición relative o absolute, de lo contrario se posicionará en relación con el cuerpo de la página.

Además, si no se especifica ningún valor de desplazamiento de la caja, por defecto el elemento será posicionado en la esquina superior izquierda de su más cercano parente con propiedad relative.

#### 4.2.- Posición Fija (fixed)

El uso de la posición fixed es similar a la posición absolute, sin embargo el posicionamiento es relativo a la ventana del navegador y no permitirá desplazamiento de la página. Dicho esto, los elementos estarán siempre presentes sin importar donde el usuario se encuentra en una página. La única advertencia es que el posicionamiento fixed no funciona con Internet Explorer 6.

Si desea forzar la posición fija dentro de Internet Explorer 6, debe usar un hack adecuado.

The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab displays the following code:

```
1 <div class="fixed">Caja Nro. 1</div>
2 <div>Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>
```

The 'CSS' tab displays the following styles:

```
1 div
2 {
3     height: 50px;
4     outline: 1px solid black;
5     width: 100px;
6 }
7 .fixed
8 {
9     left: 20px;
10    outline: 1px solid red;
11    position: fixed;
12    top: 20px;
13 }
```

Below the developer tools, the browser window shows four div elements stacked vertically. The first div, which has a red border and a red outline, is positioned at the top-left of the page, demonstrating its fixed position relative to the browser window.

## Capítulo 5. POSICIONAMIENTO DE ELEMENTO SEGÚN SU FLUJO

### 5.1.- Introducción

Cuando se trata de la construcción y posicionamiento de una página hay un puñado de diferentes técnicas a utilizar.

¿Qué técnica a utilizar?

Depende en gran medida del contenido y los objetivos de la página, ya que algunas técnicas pueden ser mejores que las demás.



Los navegadores crean y posicionan de forma automática todas las cajas que forman cada página HTML. No obstante, CSS permite al diseñador modificar la posición en la que se muestra cada caja.

El posicionamiento de una caja se establece mediante la propiedad position.

Utilizando las propiedades que proporciona CSS para alterar la posición de las cajas es posible realizar efectos muy avanzados y diseñar estructuras de páginas que de otra forma no serían posibles.

### 5.2.- Posición Normal (static)

El posicionamiento CSS es la técnica utilizada para maquetar diseños usando únicamente HTML y las propiedades CSS. Es el estándar hoy en día y todo sitio web bien maquetado debería usar esta técnica. Las ventajas son muchas, pero entre ellas destaca una mejor separación entre la apariencia y la estructura de la página, un código más semántico y entendible por buscadores, es decir, mejor para SEO, y un tamaño de página resultante inferior, por lo tanto mayor velocidad de carga de la página.

El posicionamiento normal de todos los elementos es static, que significa que cada elemento se posiciona donde le corresponde según el flujo normal de la página. Si no se indica nada, este es el valor que toman todos los elementos por defecto.

Habitualmente no se debería especificar, a menos que necesites reponer la posición normal tras haber reubicado con otra propiedad.

HTML	CSS
1 <div>Caja Nro. 1</div> 2 <div>Caja Nro. 2</div> 3 <div>Caja Nro. 3</div> 4 <div>Caja Nro. 4</div>	1 div 2 { 3   height: 50px; 4   outline: 1px solid black; 5   width: 100px; 6 }

Caja Nro. 1

Caja Nro. 2

Caja Nro. 3

Caja Nro. 4

### 5.3.- Posición Relativa (relative)

El valor relative de la propiedad position es similar al valor static.

HTML	CSS
1 <div class="relative">Caja Nro. 1</div> 2 <div>Caja Nro. 2</div> 3 <div>Caja Nro. 3</div> 4 <div>Caja Nro. 4</div>	1 div 2 { 3   height: 50px; 4   outline: 1px solid black; 5   width: 100px; 6 } 7 .relative 8 { 9   position: relative; 10 }

Caja Nro. 1

Caja Nro. 2

Caja Nro. 3

Caja Nro. 4

La diferencia es que el valor relative acepta las propiedades de desplazamiento de la caja que son: top, bottom, left y right. Estas propiedades permiten que el elemento se posicione de manera precisa a la distancia indicada respecto de donde debería estar según el flujo normal de la página y adicionalmente la posición relative puede solapar o superponer otros elementos sin moverlos de su posición por defecto.

The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab displays the following code:

```
1 <div class="relative">Caja Nro. 1</div>
2 <div>Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>
```

The 'CSS' tab displays the following styles:

```
1 div
2 {
3     height: 50px;
4     outline: 1px solid black;
5     width: 100px;
6 }
7 .relative
8 {
9     left: 20px;
10    outline: 1px solid red;
11    position: relative;
12    top: 20px;
13 }
```

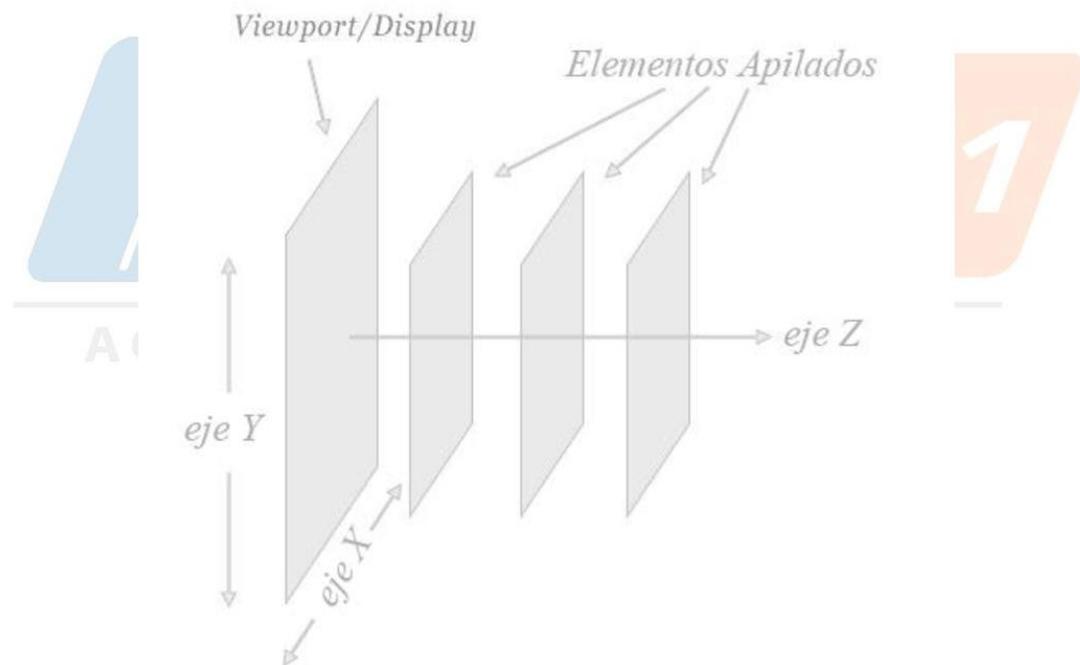
Below the code, there is a visual representation of four boxes labeled 'Caja Nro. 1' through 'Caja Nro. 4'. 'Caja Nro. 1' is a black box at the top left. 'Caja Nro. 2' is a red box positioned below and to the right of 'Nro. 1'. 'Caja Nro. 3' is a white box below 'Nro. 2'. 'Caja Nro. 4' is a black box below 'Nro. 3'. A large orange number '1' is visible on the right side of the interface.

## Capítulo 6. CONTROL DEL ORDEN DE APILAMIENTO Y COMPORTAMIENTO

### 6.1.- Apilamiento de Cajas (z-index)

Por naturaleza, las páginas web son a menudo consideradas para estar en dos dimensiones, mostrando elementos en el eje X y en el eje Y. Sin embargo, cuando se empieza a posicionar elementos, ellos ocasionalmente se posicionan uno arriba del otro.

Para cambiar el orden de como los elementos deben ser apilados, también conocido como el eje Z, la propiedad z-index nos permiten controlar el orden de apilamiento.



Generalmente, los elementos se colocan sobre el eje Z, tal como aparecen en el DOM (Document Object Model). Los elementos que vienen en la parte superior del DOM se colocan detrás de los elementos que vienen después de ellos. Cambiando el apilamiento

utilizando la propiedad z-index, lo podemos traer al frente. El elemento con el valor z-index más alto aparecerá en la parte superior, independientemente de su ubicación en el DOM.

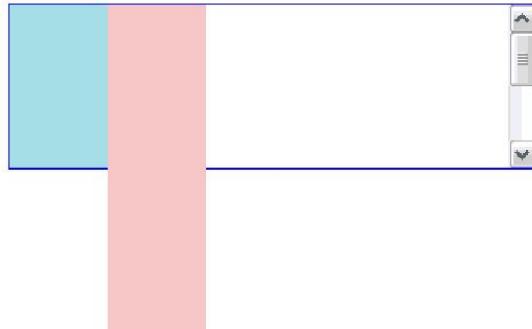
Para aplicar la propiedad z-index a un elemento, debe aplicar primero un valor de posición relative, absolute o fixed.

```
HTML
1 <div class="relative">Caja Nro. 1</div>
2 <div>Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>

CSS
1 div
2 {
3     background: #ccc;
4     height: 50px;
5     outline: 1px solid black;
6     width: 100px;
7     position: relative;
8 }
9 .relative
10 {
11     background: white;
12     z-index: 1;
13     top: 20px;
14 }
```

## 6.2.- Contenido Excedente (overflow)

Controla el comportamiento de los contenidos que no caben en su elemento contenedor.



`overflow: visible;`

Valor por defecto de la propiedad `overflow`, que es el comportamiento que hace que los contenidos se salgan del elemento y sean completamente visibles.

The screenshot shows a browser's developer tools with two tabs: "HTML" and "CSS". The "HTML" tab displays the following code:

```
1 <div>
2   <p>Lorem ipsum dolor sit amet,
3     consectetur adipiscing elit. Praesent
4     malesuada lacus vel scelerisque sodales.
5     Mauris eget magna nunc. Morbi non turpis
6     lacinia, convallis mauris et, rutrum
7     nulla. Lorem ipsum dolor sit amet,
8     consectetur adipiscing elit.</p>
9 </div>
```

The "CSS" tab displays the following styles:

```
1 div
2 {
3   height: 50px;
4   outline: 1px solid black;
5   width: 400px;
6 }
```

Below the code editor, the browser's main window shows a single `<div>` element containing a `<p>` paragraph. The paragraph's content is partially cut off at the bottom, demonstrating that the content is visible despite the container's height being less than the total content height.

`overflow: hidden;`

Hace que el navegador oculte todos los contenidos que se salen de la caja del elemento.

```

HTML
1 <div>
2   <p>Lorem ipsum dolor sit amet,
3     consectetur adipiscing elit. Praesent
4     malesuada lacus vel scelerisque sodales.
5     Mauris eget magna nunc. Morbi non turpis
6     lacinia, convallis mauris et, rutrum
7     nulla. Lorem ipsum dolor sit amet,
     consectetur adipiscing elit.</p>
3 </div>

CSS
1 div
2 {
3   height: 50px;
4   outline: 1px solid black;
5   overflow: hidden;
6   width: 400px;
7 }



... (truncated text)


```

`overflow: scroll;`

Hace que el navegador muestre barras de scroll sobre un elemento cuando sus contenidos no caben en la caja del elemento.

```

HTML
1 <div>
2   <p>Lorem ipsum dolor sit amet,
3     consectetur adipiscing elit. Praesent
4     malesuada lacus vel scelerisque sodales.
5     Mauris eget magna nunc. Morbi non turpis
6     lacinia, convallis mauris et, rutrum
7     nulla. Lorem ipsum dolor sit amet,
     consectetur adipiscing elit.</p>
3 </div>

CSS
1 div
2 {
3   height: 50px;
4   outline: 1px solid black;
5   overflow: scroll;
6   width: 400px;
7 }



... (truncated text)


```

`overflow: auto;`

Permite que cada navegador tome la decisión sobre si es necesario o no mostrar las barras de scroll.

The screenshot shows a code editor interface with two tabs: 'HTML' and 'CSS'. The 'HTML' tab contains the following code:

```
1 <div>
2   <p>Lorem ipsum dolor sit amet,
3     consectetur adipiscing elit. Praesent
4     malesuada lacus vel scelerisque sodales.
5     Mauris eget magna nunc. Morbi non turpis
6     lacinia, convallis mauris et, rutrum
7     nulla. Lorem ipsum dolor sit amet,
     consectetur adipiscing elit.</p>
3 </div>
```

The 'CSS' tab contains the following code:

```
1 div
2 {
3   height: 50px;
4   outline: 1px solid black;
5   overflow: auto;
6   width: 400px;
7 }
```

Below the tabs, there is a preview window showing the rendered HTML content:

Lore ipsum dolor sit amet, consectetur adipiscing elit.  
Praesent malesuada lacus vel scelerisque sodales. Mauris eget



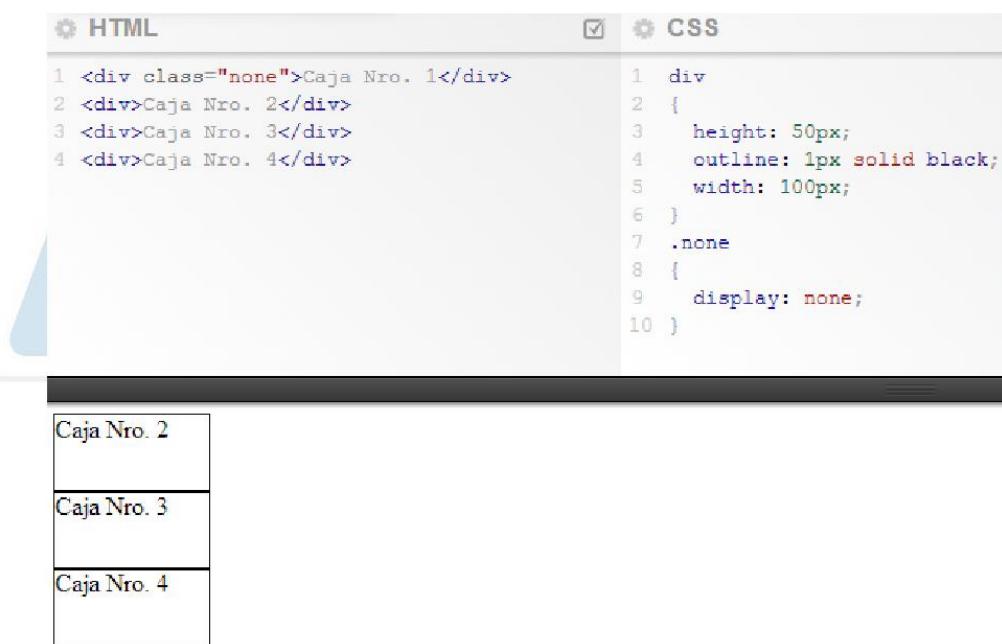
## Capítulo 7. VISUALIZACIÓN

### 7.1.- Tipo de Caja (display)

Especifica el tipo de caja que debe tener un elemento, que puede ser de diversas formas.

display: none;

Hace que el elemento no aparezca en la página ni se reserve espacio para ella.



The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab displays the following code:

```
1 <div class="none">Caja Nro. 1</div>
2 <div>Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>
```

The 'CSS' tab displays the following code:

```
1 div
2 {
3   height: 50px;
4   outline: 1px solid black;
5   width: 100px;
6 }
7 .none
8 {
9   display: none;
10 }
```

Below the developer tools, the browser window shows the visual output. It contains four boxes stacked vertically. The second box is labeled 'Caja Nro. 2', the third 'Caja Nro. 3', and the fourth 'Caja Nro. 4'. The first box is not visible because its CSS rule includes 'display: none;'.

display: block;

Hace que la caja de un elemento sea de bloque.

```

HTML checkbox CSS
1 <span class="block">Caja Nro. 1</span>
2 <span>Caja Nro. 2</span>
3 <span>Caja Nro. 3</span>
4 <span>Caja Nro. 4</span>

```

```

span
{
    height: 50px;
    outline: 1px solid black;
    width: 100px;
}
.block
{
    display: block;
}

```

Caja Nro. 1

Caja Nro. 2 Caja Nro. 3 Caja Nro. 4

**display: inline;**

Hace que la caja de un elemento sea en línea.

```

HTML checkbox CSS
1 <div>Caja Nro. 1</div>
2 <div>Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>

```

```

div
{
    display: inline;
    height: 50px;
    outline: 1px solid black;
    width: 100px;
}

```

Caja Nro. 1 Caja Nro. 2 Caja Nro. 3 Caja Nro. 4

## 7.2.- Visibilidad Del Elemento (visibility)

La propiedad **visibility** es similar a la propiedad **display** pero mucho más limitada. Mientras que **display** permite cambiar el tipo de caja de un elemento, la propiedad **visibility** sólo permite hacer visible o invisible una caja.

**visibility: visible;**

Valor por defecto de visibility, hace que la caja del elemento se vea de forma normal.

The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab contains the following code:

```
1 <div>Caja Nro. 1</div>
2 <div>Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>
```

The 'CSS' tab contains the following code:

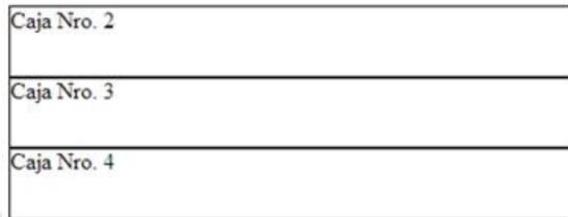
```
1 div
2 {
3   height: 50px;
4   outline: 1px solid black;
5   width: 400px;
6 }
```

Below the tabs, there are four boxes labeled 'Caja Nro. 1' through 'Caja Nro. 4', each with a black border and containing its respective text. The background features a blue and orange abstract design.

visibility: hidden;  
Hace que la caja del elemento sea invisible.

```
HTML
1 <div class="hidden">Caja Nro. 1</div>
2 <div>Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>

CSS
1 div
2 {
3   height: 50px;
4   outline: 1px solid black;
5   width: 400px;
6 }
7 .hidden
8 {
9   visibility: hidden;
10 }
```



El efecto de hacer invisible una caja mediante `visibility: hidden` es muy diferente al efecto de ocultarla mediante `display: none`.

```
HTML
1 <div class="hidden">Caja Nro. 1</div>
2 <div class="display">Caja Nro. 2</div>
3 <div>Caja Nro. 3</div>
4 <div>Caja Nro. 4</div>

CSS
1 div
2 {
3   height: 50px;
4   outline: 1px solid black;
5   width: 400px;
6 }
7 .display
8 {
9   display: none;
10 }
11 .hidden
12 {
13   visibility: hidden;
14 }
```



## Capítulo 8. SELECTORES AVANZADOS. PARTE 1

### 8.1.- Comunes (clase, id y tipo)

Los selectores son uno de, si no, las partes más importantes de CSS, conocer y dominar todos los selectores de CSS es imprescindible para crear diseños web profesionales.

Hasta hace poco el foco del CSS en realidad nunca se refirió a los selectores. De vez en cuando había actualizaciones incrementales dentro de la especificación de los selectores, pero nunca las mejoras pioneras reales. CSS3 trajo nuevos selectores, abriendo un nuevo mundo de oportunidades y mejoras en las prácticas existentes. Aquí vamos a aprender selectores, antiguos y nuevos, y cómo mejor ponerlos en práctica.



No obstante, los selectores avanzados de CSS 2.1 permiten simplificar las reglas CSS y también el código HTML, desafortunadamente, los navegadores obsoletos como Internet Explorer 6 y sus versiones anteriores no soportan este tipo de selectores avanzados, por lo que su uso no era común hasta hace poco tiempo. Hoy en día, todos los navegadores más utilizados soportan los selectores avanzados de CSS 2.1 y algunos de ellos también soportan la mayoría o todos los selectores propuestos por la futura versión CSS3.

Antes de profundizar en el tema de los selectores, y los que se ofrecen dentro de CSS3, vamos a echar un rápido vistazo a algunos de los selectores más comunes que se observan en la actualidad. Estos selectores incluyen: class (.), id (#) y tipo (etiqueta).

El selector de tipo (etiqueta) identifica un elemento basado en su tipo, específicamente como este elemento se declara dentro de HTML. El selector de clase (.) identifica un elemento basado en su valor del atributo class, que puede ser reutilizado en múltiples elementos como sea necesario para ayudar a compartir estilos comunes.

Los selectores son uno de, si no, las partes más importantes de CSS, conocer y dominar todos los selectores de CSS es imprescindible para crear diseños web profesionales.



The screenshot shows a browser window with developer tools open. On the left, the 'HTML' pane displays the following code:

```

1 <section id="sliders">
2   <h1>Tituloo</h1>
3   <p class="resaltado">Contenido</p>
4 </section>

```

On the right, the 'CSS' pane displays the following styles:

```

1 h1 { color: orange; }
2 #sliders { border-left: 1px solid black; }
3 .resaltado { color: red; }

```

Below the panes, the browser's main content area shows the rendered HTML. The title 'Tituloo' is displayed in orange, and the paragraph 'Contenido' is displayed in red.

## 8.2.- Hijos Descendientes

Como verán, hasta las hojas de estilo en cascada tienen descendencia. A través de CSS podemos reconocer las etiquetas que en la estructura HTML pertenecen o están

contenidas dentro de otra etiqueta, y aplicarles estilos determinados. En CSS estos se llaman Selectores Descendentes.

Esta selección puede ser hecha de dos diferentes manera, usando descendiente o selectores hijos directamente.

### Selectores hijos descendiente

Se ocupan para afectar a todos los elementos que pertenecen (o descienden) de una determinada etiqueta padre. El elemento descendiente no tiene que venir directamente después del elemento ancestro dentro de la estructura del documento, como una relación padre-hijo, pero puede estar en cualquier lugar dentro del elemento ancestro.

Los selectores descendientes son creados con una separación de elementos dentro de un selector, creando un nuevo nivel de jerarquía para cada lista de elementos.

The diagram illustrates the relationship between HTML and CSS. On the left, under 'HTML', is the following code:

```
1 <section>
2   <h2>Etiqueta H2</h2>
3   <article>
4     <h2>Etiqueta H2</h2>
5     <p>Párrafo</p>
6   </article>
7 </section>
```

On the right, under 'CSS', is the following style:

```
1 section h2
2 {
3   color: green;
4 }
```

**Etiqueta H2**

**Etiqueta H2**

Párrafo

### 8.3.- Hijos directos

Muchas veces los selectores descendientes seleccionan más de lo esperado. A veces solo los hijos directos de un elemento primario deben ser seleccionados y no cada instancia del elemento anidada profundamente en el interior de un antepasado.

Se trata de un selector similar al selector descendente, pero muy diferente en su funcionamiento. Se utiliza para seleccionar un elemento que es hijo de otro elemento y se indica mediante el "signo de mayor que" (>).

Mientras que en el selector descendente sólo importa que un elemento esté dentro de otro, independientemente de lo profundo que se encuentre, en el selector de hijos el elemento debe ser hijo directo de otro elemento. No soportado por Internet Explorer 6.



The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab displays the following code:

```
1 <section>
2   <h2>Etiqueta H2</h2>
3   <article>
4     <h2>Etiqueta H2</h2>
5     <p>Párrafo</p>
6   </article>
7 </section>
```

The 'CSS' tab contains the following rule:

```
1 section > h2
2 {
3   color: green;
4 }
```

**Etiqueta H2**

**Etiqueta H2**

Párrafo

## Capítulo 9. SELECTORES AVANZADOS. PARTE 2

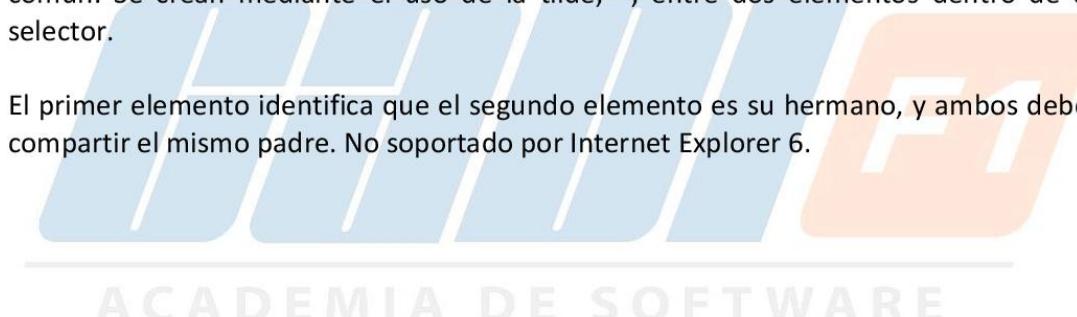
### 9.1.- Introducción

Saber elegir hijos de un elemento es una gran medida beneficiosa, y muy comúnmente visto. Sin embargo los elementos hermanos, aquellos elementos que comparten un mismo parente común, también es necesario saberlo seleccionar. Estas selecciones de hermanos se pueden hacer por medio del hermano general y de hermanos adyacentes.

### 9.2.- Hermanos

- Selectores hermanos (General): el selector de hermano general permitir seleccionar elementos en base a sus elementos relacionados, los que comparten el mismo parente común. Se crean mediante el uso de la tilde, ~, entre dos elementos dentro de un selector.

El primer elemento identifica que el segundo elemento es su hermano, y ambos deben compartir el mismo parente. No soportado por Internet Explorer 6.



The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab displays the following code:

```
1 <section>
2   <p>Párrafo</p>
3   <article>
4     <h2>Etiqueta H2</h2>
5     <p>Párrafo</p>
6     <div>
7       <p>Párrafo</p>
8     </div>
9     <p>Párrafo</p>
10   </article>
11 </section>
```

The 'CSS' tab contains the following rule:

```
1 h2 + p
2 {
3   color: red;
4 }
```

Below the code, the browser's main content area shows the rendered HTML. It features a section with three paragraphs. The first paragraph is in black text, while the second and third paragraphs are in red text, demonstrating the effect of the CSS selector.

### 9.3.- Hermanos adyacentes

- Selectores hermanos adyacentes: de vez en cuando un poco más de control puede ser deseable, incluyendo la capacidad para seleccionar un elemento adyacente que sigue directamente después de otro elemento adyacente. Ahí es donde el elemento hermano adyacente entra en juego. El selector de hermanos adyacentes sólo selecciona elementos del mismo nivel que sigue directamente después de otro elemento adyacente.

Son los que comparten la misma etiqueta padre y que se encuentran uno a lado del otro en la estructura HTML. Se identifica con un signo + entre los hermanos, y la etiqueta que tendrá el estilo será la última declarada luego de este +.

HTML	CSS
<pre>1 &lt;section&gt; 2   &lt;p&gt;Párrafo&lt;/p&gt; 3   &lt;article&gt; 4     &lt;h2&gt;Etiqueta H2&lt;/h2&gt; 5     &lt;p&gt;Párrafo&lt;/p&gt; 6     &lt;div&gt; 7       &lt;p&gt;Párrafo&lt;/p&gt; 8     &lt;/div&gt; 9     &lt;p&gt;Párrafo&lt;/p&gt; 10    &lt;/article&gt; 11 &lt;/section&gt;</pre>	<pre>1 h2 + p 2 { 3   color: red; 4 }</pre>

Párrafo

## Etiqueta H2

Párrafo

Párrafo

Párrafo

ACADEMIA DE SOFTWARE

## Capítulo 10. SELECTORES AVANZADOS SEGÚN EL ATRIBUTO

### 10.1.- Introducción

Permiten seleccionar elementos HTML en función de sus atributos y/o valores de esos atributos.

### 10.2.- Atributo presente

Selector de atributo presente: selecciona un elemento si el atributo especificado está presente.

```
HTML
1 <a href="#" class="link">cadif1</a>
2 <a href="#" class="link2">cadif1</a>
3 <a href="#">cadif1</a>

CSS
1 a[class]
2 {
3   color: red;
4 }
```

cadif1 cadif1 cadif1

### 10.3.- Atributo igual

Selector de atributos igual: selecciona un elemento si el valor del atributo especificado coincide exactamente con el valor indicado.

```
HTML
1 <a href="#" class="link">cadif1</a>
2 <a href="#" class="link2">cadif1</a>
3 <a href="#">cadif1</a>

CSS
1 a[class="link2"]
2 {
3   color: red;
4 }
```

cadif1 cadif1 cadif1

#### 10.4.- Atributo contenido

Selector de atributos contenido: selecciona un elemento si el valor del atributo especificado contiene al menos una vez por instancia del valor declarado.

```
HTML
1 <a href="cadif1.html">cadif1</a>
2 <a href="#">cadif1</a>
3 <a href="contacto/cadif1.html">cadif1</a>

CSS
1 a[href*="cadif1"]
2 {
3   color: red;
4 }
```

cadif1 cadif1 cadif1

## Capítulo 11. SELECTORES CON PSEUDO-CLASES

### 11.1.- Pseudo-clases

Muchos elementos del lenguaje de marcas HTML disponen de estados especiales o usos asociados a ellos. Podemos aplicar estilos diferentes a estos elementos basándonos en esos estados o usos en nuestras hojas de estilo CSS. Por ejemplo, el elemento a tiene cuatro estados:

- link: su estado normal.
- visited: cuando ya hemos visitado el link al que hace referencia.
- hover: cuando tenemos el cursor situado encima del mismo.
- focus: selecciona enlaces en los que se encuentra posicionado el cursor del teclado en el momento actual.
- active: cuando hacemos click sobre él.

```
1 <a href="#">cadif1</a>
2 <a href="#">cadif1</a>
3 <a href="#">cadif1</a>
```

```
1 a:link { color: blue; }
2 a:visited { color: gray; }
3 a:hover, a:focus { text-decoration: none; }
4 a:active { font-weight: bold; }
```

### 11.2.- Estructura y posición

Un puñado de pseudo-clases son estructurales y basada en la posición, en las que se determinan en función de los elementos que residen en la estructura del documento. Cada uno de los cuales proporciona su propia función única diferente. Algunas pseudo-clases han estado por más tiempo que otras, sin embargo CSS3 trajo toda una nueva serie de pseudo-clases para complementar las ya existentes.

-:first-child: fija el aspecto del primer elemento de un tipo de selector si es el primer nodo hijo de su elemento padre

The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab displays the following code:

```
1 <div>
2   <p>HTML5</p>
3   <p>CSS3</p>
4   <p>Pre-procesadores</p>
5   <p>jQuery</p>
6   <p>JSON</p>
7 </div>
```

The 'CSS' tab contains the following styles:

```
1 p:first-child
2 {
3   font-weight: bold;
4   text-decoration: line-through;
5 }
```

Below the tabs, there is a preview area showing the rendered HTML. The first 'p' element ('HTML5') is bolded and has a line-through effect. The other five 'p' elements ('CSS3', 'Pre-procesadores', 'jQuery', 'JSON') are regular text.

On the right side of the preview area, there is a decorative graphic consisting of blue and orange curved shapes.

Below the preview area, there is a list of terms:

- HTML5
- CSS3
- Pre-procesadores
- jQuery
- JSON

A note at the bottom left of the preview area states: "-:last-child: fija el aspecto del último hijo del tipo de selector de un elemento padre."

---

The second part of the screenshot shows the same setup but with the 'HTML' tab selected. The code remains the same:

```
1 <div>
2   <p>HTML5</p>
3   <p>CSS3</p>
4   <p>Pre-procesadores</p>
5   <p>jQuery</p>
6   <p>JSON</p>
7 </div>
```

The 'CSS' tab contains the same styles as before:

```
1 p:last-child
2 {
3   font-weight: bold;
4   text-decoration: line-through;
5 }
```

The preview area shows that the last 'p' element ('JSON') is bolded and has a line-through effect, while the others are regular text.

---

Below the preview area, there is a list of terms:

- HTML5
- CSS3
- Pre-procesadores
- jQuery
- JSON

`:first-of-type`: fija el aspecto del primer elemento de un tipo de selector si es el primer nodo hijo que aparece con su tipo de selector en su elemento padre (esto parece complejo pero en los ejemplos veremos que no lo es).

```
❶ <div>
❷   <h1>Diseño Web Profesional</h1>
❸   <p>HTML5</p>
❹   <p>CSS3</p>
❺   <p>Pre-procesadores</p>
❻   <p>jQuery</p>
❼   <p>JSON</p>
❽   <a href="cadif1.com">cadif1</a>
❾ </div>
```

```
❶ p:first-of-type
❷ {
❸   text-decoration: underline;
❹ }
```

**Diseño Web Profesional**

HTML5

CSS3

Pre-procesadores

jQuery

JSON

[cadif1](cadif1.com)

`:last-of-type`: fija el aspecto del último hijo del tipo de selector de un elemento padre.

HTML	CSS
<pre>1 &lt;div&gt; 2   &lt;h1&gt;Diseño Web Profesional&lt;/h1&gt; 3   &lt;p&gt;HTML5&lt;/p&gt; 4   &lt;p&gt;CSS3&lt;/p&gt; 5   &lt;p&gt;Pre-procesadores&lt;/p&gt; 6   &lt;p&gt;jQuery&lt;/p&gt; 7   &lt;p&gt;JSON&lt;/p&gt; 8   &lt;a href="cadif1.com"&gt;cadif1&lt;/a&gt; 9 &lt;/div&gt;</pre>	<pre>1 p:last-of-type 2 { 3   text-decoration: underline; 4 }</pre>

## Diseño Web Profesional

[HTML5](#)

[CSS3](#)

[Pre-procesadores](#)

[jQuery](#)

[JSON](#)

[cadif1](#)



**ACADEMIA DE SOFTWARE**  
-:nth-child(#): fija el aspecto de una ocurrencia específica del elemento nodo hijo especificado. Por ejemplo: el tercer elemento nodo hijo de una lista no ordenada sería li::nth-child(3). Además se pueden usar pequeñas expresiones como parámetro para por ejemplo, seleccionar todos los elementos impares: li::nth-child(2n+1); los pares: li::nth-child(2n).

Los elementos impares también pueden ser seleccionados con li::nth-child(odd).

HTML	CSS
<pre>1 &lt;ul&gt; 2   &lt;h1&gt;Diseño Web Profesional&lt;/h1&gt; 3   &lt;li&gt;HTML5&lt;/li&gt; 4   &lt;li&gt;CSS3&lt;/li&gt; 5   &lt;ul&gt; 6     &lt;li&gt;Básico&lt;/li&gt; 7     &lt;li&gt;Avanzado&lt;/li&gt; 8   &lt;/ul&gt; 9   &lt;li&gt;Pre-procesadores&lt;/li&gt; 10  &lt;li&gt;jQuery&lt;/li&gt; 11  &lt;li&gt;JSON&lt;/li&gt; 12 &lt;/ul&gt;</pre>	<pre>1 li:nth-child(2) 2 { 3   color: red; 4 }</pre>

## Diseño Web Profesional

- HTML5
- CSS3
  - Básico
  - Avanzado
- Pre-procesadores
- jQuery
- JSON

**ACADEMIA DE SOFTWARE**  
-:nth-of-type(#): fija la apariencia de una ocurrencia específica del elemento con el tipo de selector especificado en un elemento padre. Por ejemplo: la segunda lista no ordenada sería ul:nth-of-type(2). También permite los mismos parámetros que :nth-child(#).

```
HTML
1 <ul>
2   <h1>Diseño Web Profesional</h1>
3   <li>HTML5</li>
4   <li>CSS3</li>
5   <ul>
6     <li>Básico</li>
7     <li>Avanzado</li>
8   </ul>
9   <li>Pre-procesadores</li>
10  <li>jQuery</li>
11  <li>JSON</li>
12 </ul>

CSS
1 li:nth-of-type(2)
2 {
3   color: red;
4 }
```

## Diseño Web Profesional

- HTML5
- CSS3
  - Básico
  - Avanzado
- Pre-procesadores
- jQuery
- JSON

`:nth-last-of-type(#)`: fija el aspecto de una ocurrencia específica del elemento con el tipo de selector especificado pero desde abajo. Por ejemplo: la octava lista no ordenada empezando por abajo sería `ul:nth-last-of-type(8)`. También le es aplicable la parametrización de `:nth-child(#)`.

```
 HTML  CSS
```

```
1 <ul>
2   <h1>Diseño Web Profesional</h1>
3   <li>HTML5</li>
4   <li>CSS3</li>
5   <ul>
6     <li>Básico</li>
7     <li>Avanzado</li>
8   </ul>
9   <li>Pre-procesadores</li>
10  <li>jQuery</li>
11  <li>JSON</li>
12 </ul>
```

## Diseño Web Profesional

- HTML5
- CSS3
  - Básico
  - Avanzado
- Pre-procesadores
- jQuery
- JSON

ACADEMIA DE SOFTWARE

-:not: para no dar un estilo específico a un selector

```
 HTML  CSS
```

```
1 <ul>
2   <h1>Diseño Web Profesional</h1>
3   <li class="home">HTML5</li>
4   <li>CSS3</li>
5   <ul>
6     <li>Básico</li>
7     <li>Avanzado</li>
8   </ul>
9   <li>Pre-procesadores</li>
10  <li>jQuery</li>
11  <li>JSON</li>
12 </ul>
```

## Diseño Web Profesional

- HTML5
- CSS3
  - Básico
  - Avanzado
- Pre-procesadores
- jQuery
- JSON

ACADEMIA DE SOFTWARE

## Capítulo 12. EFECTOS ESPECIALES. PARTE 1

### 12.1.- Border-radius

CSS3 trae una serie de novedades en el manejo de bordes de elementos.

Hasta ahora era complicado y una pérdida de tiempo crear un borde redondeado, teniendo que utilizar imágenes de fondo.

El atributo border-radius simplifica su implementación.



The screenshot shows the browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab displays the following code:

```
1 <section>
2   <article>
3     <p>Border Radius</p>
4   </article>
5 </section>
```

The 'CSS' tab displays the following CSS code:

```
1 section
2 {
3   border-radius: 1em;
4   border: 1px solid gray;
5   height: 50px;
6   padding: 1em;
7   width: 200px;
8 }
```

Below the developer tools, there is a visual representation of a rounded rectangle with the text "Border Radius" inside it.

Como resultado tendríamos una capa con todas las esquinas redondeadas.

Si quisieramos usar diferentes radios en cada esquina, la sintaxis shorthand sería:

```

 HTML
<section>
  <article>
    <p>Border Radius</p>
  </article>
</section>

 CSS
1 section
2 {
3   border-radius: 1em .5em 2em 1.5em;
4   border: 1px solid gray;
5   height: 50px;
6   padding: 1em;
7   width: 200px;
8 }

```

## 12.2.- Box-shadow

Hasta ahora aplicar un efecto de sombra a cualquier elemento de nuestro HTML, era un proceso donde teníamos que cargar imágenes creadas previamente en nuestro programa de edición de imágenes, recortales, ajustarlas, etc.

Con el atributo box-shadow podemos aplicar sombras a nuestras capas con mucha facilidad.

La sintaxis de box-shadow es la siguiente:

`box-shadow: distanciaX distanciaY difuminado color;`

```

 HTML
1 <div>
2   <h1>Box-Shadow</h1>
3 </div>

 CSS
1 div
2 {
3   box-shadow: 5px 10px 7px rgb(200,200,200);
4   height: 50px;
5   width: 200px;
6 }

```

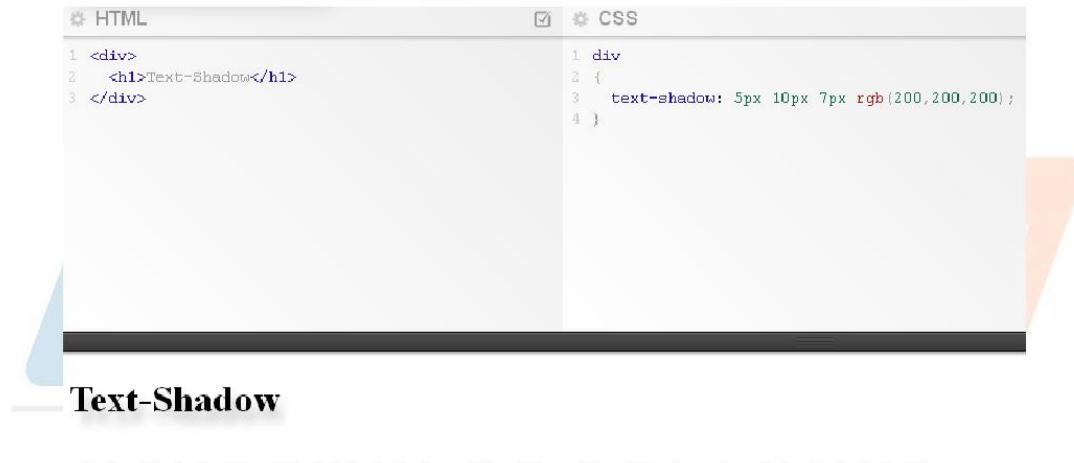
## Box-Shadow

### 12.3.- Text-shadow

El atributo text-shadow se incluyó inicialmente en el CSS2 pero fue eliminado en CSS2.1. Por suerte para todos CSS3 vuelve a incluirlo.

text-shadow realiza una sombra ajustada a los propios caracteres de un texto.

La sintaxis de text-shadow es igual a la de box-shadow: text-shadow: distanciaX distanciaY difuminado color;



The screenshot shows a browser's developer tools with two tabs: "HTML" and "CSS". The "HTML" tab contains the following code:

```
1 <div>
2   <h1>Text-Shadow</h1>
3 </div>
```

The "CSS" tab contains the following rule:

```
1 div
2 {
3   text-shadow: 5px 10px 7px #rgb(200,200,200);
4 }
```

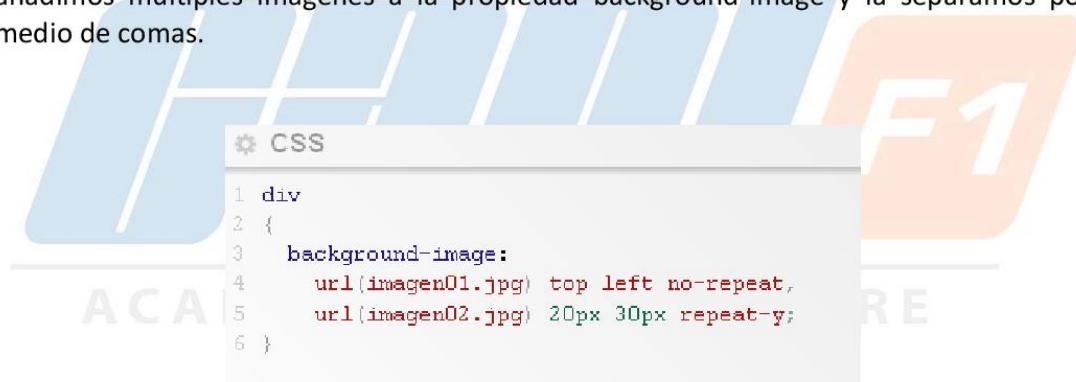
Below the developer tools, the rendered text "Text-Shadow" is displayed on a page with a blue header bar. The text has a visible orange shadow effect.

## Capítulo 13. EFECTOS ESPECIALES. PARTE 2

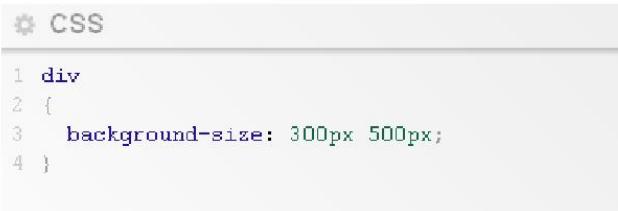
### 13.1.- Multiple background

Css3 nos trae nuevas formas de controlar los background o imágenes de fondo de cajas, div o de sitios web completos. Utilizando CSS3 podemos colocar múltiples imágenes de fondo a un elemento de html podría ser un div. Antes para conseguir esto teníamos que crear una sola composición con todas las imágenes o múltiples divs para conseguir este objetivo, con las nuevas y mejoradas propiedades de background de css3 conseguiremos ajustarlos a la necesidad presentada.

- **background-images:** nos va a permitir colocar varias imágenes de fondo. La declaración de la propiedad es la misma de la anterior versión de css, solo que ahora añadimos múltiples imágenes a la propiedad background-image y la sepáramos por medio de comas.



- **background-size:** por medio de esta propiedad vamos a poder ajustar o escalar nuestro background o imagen de fondo de nuestro sitio web, podemos ajustar el ancho y alto, podemos ajustarla y adaptarla a la resolución de la pantalla del navegador o div contenedor.



```
1 div
2 {
3     background-size: 300px 500px;
4 }
```

El tamaño también lo podemos definir por medio de porcentaje, longitud, cover y contain.

- porcentaje: establecemos el tamaño de la imagen dependiendo de su del elemento padre.





- cover: escala la imagen para el tamaño más pequeño de tal manera que su anchura y su altura puede caber dentro del área de contenido.

```
* CSS
1 div
2 {
3   background-size: cover;
4 }
```

- contain: escala la imagen al tamaño más grande de tal manera que su anchura y su altura puede caber dentro del área de contenido

```
* CSS
1 div
2 {
3   background-size: contain;
4 }
```

- background-origin: esta propiedad nos permite posicionar el background o imagen de fondo a relación del contenido o area de la caja contenedora, podemos hacer que sea relativa al padding de la caja por medio de padding-box, también por medio de el borde de la caja por medio de border-box y también la podemos ubicar relativamente a el contenido de la caja por medio de content-box.



## Capítulo 14. TRANSFORMACIÓN DE ELEMENTO

### 14.1.- Transformaciones

A partir de ahora podremos modificar la rotación, inclinación o escala de nuestros elementos.

El atributo transform nos permite, como su propio nombre indica, transformar un elemento. Su sintaxis es la siguiente: transform: tipo(cantidad);. El valor tipo puede tomar cuatro valores, y cada uno de ellos realiza una función diferente.

- rotate: nos permite girar los elementos en número de grados.



The screenshot shows a browser's developer tools with two tabs: 'HTML' and 'CSS'. The 'HTML' tab contains the code <div></div>. The 'CSS' tab contains the following CSS:

```
1 div
2 {
3   border: 1px solid black;
4   height: 150px;
5   margin: 2em;
6   transform: rotate(10deg);
7   width: 200px;
8 }
```

Below the tabs, the browser window displays a single red-bordered div element that has been rotated 10 degrees clockwise.

- Skew: podemos inclinar un elemento tanto en coordenadas X como Y, el valor se expresa en grados.

The screenshot shows the browser's developer tools with two tabs: "HTML" and "CSS". The "HTML" tab contains the code: 

```
1 <div></div>
```

. The "CSS" tab contains the following CSS rules:

```
1 div
2 {
3   border: 1px solid black;
4   height: 150px;
5   margin: 2em;
6   transform: skew(10deg, 5deg);
7   width: 200px;
8 }
```

A preview window below shows a white rectangular box with its top-left corner skewed upwards and to the right.

- **scale:** nos permite escalar elementos tanto en el eje X como en el eje Y en una cantidad expresada en tantos por uno.

The screenshot shows the browser's developer tools with two tabs: "HTML" and "CSS". The "HTML" tab contains the code: 

```
1 <div></div>
```

. The "CSS" tab contains the following CSS rules:

```
1 div
2 {
3   border: 1px solid black;
4   height: 150px;
5   margin: 2em;
6   transform: scale(.7, .8);
7   width: 200px;
8 }
```

A preview window below shows a white rectangular box that is smaller than the original, indicating a scaling effect.

- translate: podemos desplazar el elemento tanto en el eje X y como en el eje Y.

The screenshot shows a browser's developer tools with two tabs: "HTML" and "CSS". The "HTML" tab contains the code: 

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

. The "CSS" tab contains the following code:

```
1 div
2 {
3   border: 1px solid black;
4   height: 150px;
5   margin: 2em;
6   transform: translate(20px, 10px);
7   width: 200px;
8 }
```

Below the developer tools, there is a visual representation of a white square with a black border, which has been translated 20 pixels horizontally and 10 pixels vertically from its original position.

Se puede aplicar diferentes transformaciones a un mismo elemento simplemente escribiéndoles de manera consecutiva:

The screenshot shows a browser's developer tools with two tabs: "HTML" and "CSS". The "HTML" tab contains the code: 

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

. The "CSS" tab contains the following code:

```
1 div
2 {
3   border: 1px solid black;
4   height: 150px;
5   margin: 2em;
6   transform:
7     rotate(17deg)
8     scale(0.576)
9     skew(32deg)
10    translate(34px);
11   width: 200px;
12 }
```

Below the developer tools, there is a visual representation of a white pentagon with a black border, which has been rotated, scaled, skewed, and translated from its original position.

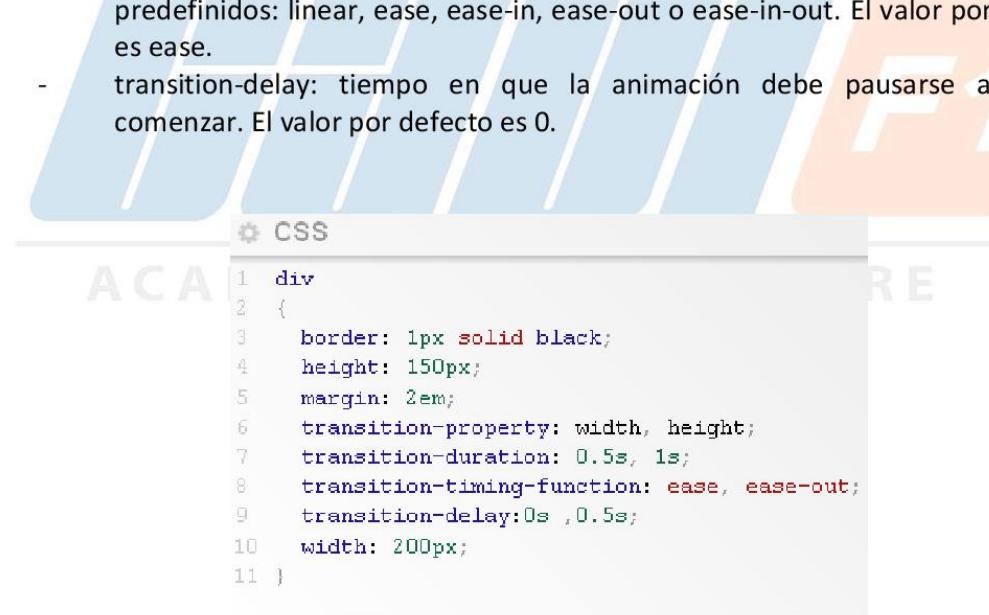
## Capítulo 15. TRANSICIONES DE ELEMENTOS

### 15.1.- Transiciones

Actualmente CSS3 permite la posibilidad de cambiar el estilo de un elemento mediante una transición animada.

Existen 4 propiedades base para crear transiciones:

- transition-property: determina la(s) propiedad(es) a ser animadas; puede ser cualquier propiedad CSS
- transition-duration: indica la duración de la animación del inicio al fin en segundos. Por defecto el valor es 0.
- transition-timing-function: definen física de animación. Puede tener 5 valores predefinidos: linear, ease, ease-in, ease-out o ease-in-out. El valor por defecto es ease.
- transition-delay: tiempo en que la animación debe pausarse antes de comenzar. El valor por defecto es 0.

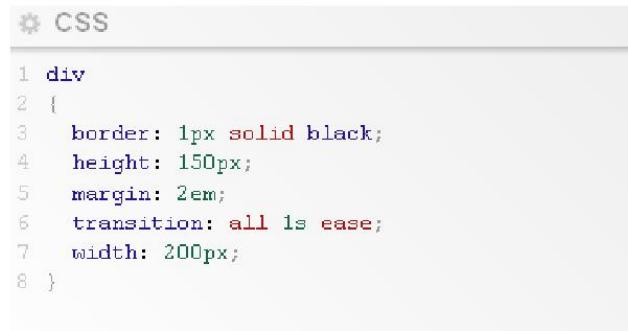


ACA RE

**CSS**

```
1 div
2 {
3     border: 1px solid black;
4     height: 150px;
5     margin: 2em;
6     transition-property: width, height;
7     transition-duration: 0.5s, 1s;
8     transition-timing-function: ease, ease-out;
9     transition-delay: 0s, 0.5s;
10    width: 200px;
11 }
```

Se puede aplicar el método shorthand (abreviado), independientemente del orden.



```
1 div
2 {
3     border: 1px solid black;
4     height: 150px;
5     margin: 2em;
6     transition: all 1s ease;
7     width: 200px;
8 }
```

## 15.2.- Prefijos Css en Los Navegadores

Los fabricantes de navegadores constantemente están desarrollando e implementando funcionalidades que forman parte de los estándares web o que encuentran de utilidad y creen que formarán parte de estos estándares en un futuro.

En este segundo caso implementan la funcionalidad, pero estableciendo a la propiedad un prefijo para remarcar su carácter no "estándar oficial" (hasta puede estar en nivel "recomendación" del estándar) y por indicar que posibles futuras adaptaciones podrían cambiar (un poco) su conducta.

### Tipos de Prefijos

-webkit-: para los navegadores basados en webkit (Chrome, Konqueror, Safari, en breve Ópera y otros).

-o-: para Ópera hasta la fecha.

-ms-: para Internet Explorer.

-moz-: navegadores basados en Gecko (Firefox y otros)

Uso

Los prefijos se anteponen al nombre de la propiedad del estilo para que el navegador web reconozca esa determinada característica. Se aplican normalmente a estilos css3 en experimentación que aún no forman parte de una especificación o estándar.

El uso de los prefijo se realiza mediante la sintaxis: -prefijo-propiedad.

Así podemos encontrar:

### Consideraciones

En el uso de prefijos css, se debe tener en cuenta que:

- Se debería indicar la propiedad genérica sin prefijo para el caso en que que el navegador no esté entre los que tienen prefijo especificado, este tenga posibilidad de interpretar el estilo.
- Se recomienda que la última de las propiedades sea la que cumplirá el estándar, y que interpretarán todos los navegadores, ya que es mucho mejor siempre usar la opción definida como estándar, que una versión propia del navegador que puede tener deficiencias o funcionalidades extras.
- El uso de estos prefijos no es estándar y provocará errores en los validadores de código css, cosa a tener en cuenta para disponer de un código limpio y por lo que debería haber un mantenimiento de las hojas de estilo, eliminando estas entradas una vez se hayan estandarizado e implementado en su versión no-específica.