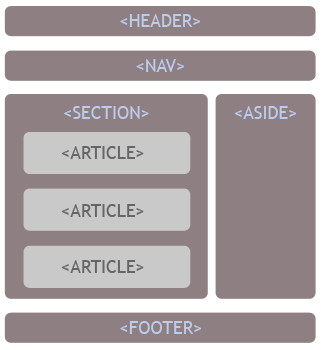
## **Qué son etiquetas semánticas**

***Semántica:****Perteneciente o relativo a la significación de las palabras.* Esa es la definición de la Real Academia de la Lengua Española y es así de sencillo es entender este tipo de etiquetas del lenguaje HTML: Aquellas que dan un significado a las partes del documento.

Por decirlo de otra forma, estamos ante **etiquetas que indican qué es el contenido que contienen, en lugar de cómo se debe formatear**al mostrar el documento HTML en un cliente web.



**Clear**

Ejemplo con clear

<!DOCTYPE html>

<html>

<head>

<title>Clear</title>

<style type="text/css">

div{

width: 150px;

height: 150px;

border: 1px solid black;

text-align: center;

}

#uno{

background: red;

float: right;

}

#dos{

background: yellow;

float: right;

}

#tres{

background: blue;

clear: right;

float: right;

}

header{

border: 1px solid black;

height: 304px;

}

</style>

</head>

<body>

<header>

<div id="uno">1</div>

<div id="dos">2</div>

<div id="tres">3</div>

</header>

</body>

</html>

**Posicionamiento**

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.

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.

El estándar de CSS define cinco modelos diferentes para posicionar una caja:

* Posicionamiento normal o estático: se trata del posicionamiento que utilizan los navegadores si no se indica lo contrario.
* Posicionamiento relativo: variante del posicionamiento normal que consiste en posicionar una caja según el posicionamiento normal y después desplazarla respecto de su posición original.
* Posicionamiento absoluto: la posición de una caja se establece de forma absoluta respecto de su elemento contenedor y el resto de elementos de la página ignoran la nueva posición del elemento.
* Posicionamiento fijo: variante del posicionamiento absoluto que convierte una caja en un elemento inamovible, de forma que su posición en la pantalla siempre es la misma independientemente del resto de elementos e independientemente de si el usuario sube o baja la página en la ventana del navegador.
* Posicionamiento flotante: se trata del modelo más especial de posicionamiento, ya que desplaza las cajas todo lo posible hacia la izquierda o hacia la derecha de la línea en la que se encuentran.

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

| **Propiedad** | **position** |
| --- | --- |
| **Valores** | static | relative | absolute | fixed | [inherit](http://librosweb.es/referencia/css/el_valor_inherit.html) |
| **Se aplica a** | Todos los elementos |
| **Valor inicial** | static |
| **Descripción** | Selecciona el posicionamiento con el que se mostrará el elemento |

El significado de cada uno de los posibles valores de la propiedad position es el siguiente:

* static: corresponde al posicionamiento normal o estático. Si se utiliza este valor, se ignoran los valores de las propiedades top, right, bottom y left que se verán a continuación.
* relative: corresponde al posicionamiento relativo. El desplazamiento de la caja se controla con las propiedades top, right, bottom y left.
* absolute: corresponde al posicionamiento absoluto. El desplazamiento de la caja también se controla con las propiedades top, right, bottom y left, pero su interpretación es mucho más compleja, ya que el origen de coordenadas del desplazamiento depende del posicionamiento de su elemento contenedor.
* fixed: corresponde al posicionamiento fijo. El desplazamiento se establece de la misma forma que en el posicionamiento absoluto, pero en este caso el elemento permanece inamovible en la pantalla.

La propiedad position no permite controlar el posicionamiento flotante, que se establece con otra propiedad llamada float y que se explica más adelante. Además, la propiedad position sólo indica cómo se posiciona una caja, pero no la desplaza.

Normalmente, cuando se posiciona una caja también es necesario desplazarla respecto de su posición original o respecto de otro origen de coordenadas. CSS define cuatro propiedades llamadas top, right,bottom y left para controlar el desplazamiento de las cajas posicionadas:

| **Propiedades** | **top**, **right**, **bottom**, **left** |
| --- | --- |
| **Valores** | [unidad de medida](http://librosweb.es/referencia/css/unidades_de_medida.html) | [porcentaje](http://librosweb.es/referencia/css/porcentajes.html) | auto | [inherit](http://librosweb.es/referencia/css/el_valor_inherit.html) |
| **Se aplica a** | Todos los elementos posicionados |
| **Valor inicial** | auto |
| **Descripción** | Indican el desplazamiento horizontal y vertical del elemento respecto de su posición original |

En el caso del posicionamiento relativo, cada una de estas propiedades indica el desplazamiento del elemento desde la posición original de su borde superior/derecho/inferior/izquierdo. Si el posicionamiento es absoluto, las propiedades indican el desplazamiento del elemento respecto del borde superior/derecho/inferior/izquierdo de su primer elemento padre posicionado.

En cualquiera de los dos casos, si el desplazamiento se indica en forma de porcentaje, se refiere al porcentaje sobre la anchura (propiedades right y left) o altura (propiedades top y bottom) del elemento.

**Propiedad z-index**

Además de posicionar una caja de forma horizontal y vertical, CSS permite controlar la posición tridimensional de las cajas posicionadas. De esta forma, es posible indicar las cajas que se muestran delante o detrás de otras cajas cuando se producen solapamientos.

La posición tridimensional de un elemento se establece sobre un tercer eje llamado Z y se controla mediante la propiedad z-index. Utilizando esta propiedad es posible crear páginas complejas con varios niveles o capas.

A continuación se muestra la definición formal de la propiedad z-index:

| **Propiedad** | **z-index** |
| --- | --- |
| **Valores** | auto | numero | [inherit](http://librosweb.es/referencia/css/el_valor_inherit.html) |
| **Se aplica a** | Elementos que han sido posicionados explícitamente |
| **Valor inicial** | auto |
| **Descripción** | Establece el nivel tridimensional en el que se muestra el elemento |

El valor más común de la propiedad z-index es un número entero. Aunque la especificación oficial permite los números negativos, en general se considera el número 0 como el nivel más bajo.

Cuanto más alto sea el valor numérico, más cerca del usuario se muestra la caja. Un elemento con z-index: 10 se muestra por encima de los elementos con z-index: 8 o z-index: 9, pero por debajo de elementos con z-index: 20 o z-index: 50.

**Ejemplo posición relativa**

<!DOCTYPE html>

<html>

<head>

<title>Posicionamiento Relativo</title>

<style type="text/css">

section{

width: 500px;

height: 400px;

border: 5px dashed blue;

margin: auto;

padding: 15px;

}

#uno{

background: rgba(0,0,255,0.5);

text-align: center;

padding:20px;

position: relative;

top:300px;

left: 15px;

}

</style>

</head>

<body>

<section>

<p>orem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod

tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo

consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse

cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non

proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

<div id="uno">1</div>

<p>orem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod

tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo

consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse

cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non

proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

</section>

</body>

</html>

**Ejemplo posición absoluta**

<!DOCTYPE html>

<html>

<head>

<title>Posicionamiento absoluto</title>

<style type="text/css">

section{

border: 5px dashed blue;

margin:80px auto;

padding: 15px;

position: relative;

}

#uno{

background: rgba(0,0,255,0.5);

text-align: center;

padding:20px;

position: absolute;

top: 0px;

left: 0px;

}

</style>

</head>

<body>

<section>

<p>orem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod

tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo

consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse

cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non

proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

<div id="uno">1</div>

<p>orem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod

tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo

consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse

cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non

proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

</section>

</body>

</html>

**Ejemplo posición fixed**

<!DOCTYPE html>

<html>

<head>

<title>Posicionamiento Fixed</title>

<style type="text/css">

section{

border: 5px dashed blue;

margin:80px auto;

padding: 15px;

padding-bottom: 500px;

}

#uno{

background: rgba(0,0,255,0.5);

text-align: center;

padding:20px;

position: fixed;

top: 0px;

left: 0px;

}

</style>

</head>

<body>

<section>

<p>orem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod

tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo

consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse

cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non

proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

<div id="uno">1</div>

<p>orem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod

tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo

consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse

cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non

proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>

</section>

</body>

</html>

**Ejemplo z-index**

<!DOCTYPE html>

<html>

<head>

<title>Z-index</title>

<style type="text/css">

section{

width: 500px;

height: 400px;

border: 5px dashed blue;

margin: auto;

padding: 15px;

}

#uno{

background: palegreen;

text-align: center;

width: 100px;

height:100px;

line-height: 100px;

top:50px;

left: 50px;

z-index: 1;

}

#dos{

background: tomato;

text-align: center;

width: 100px;

height:100px;

line-height: 100px;

top:10px;

left: 100px;

z-index: 1;

}

#Tres{

background: lightblue;

text-align: center;

width: 100px;

height:100px;

line-height: 1px;

top:-50px;

left: 150px;

z-index: 1;

}

div{

border-radius: 5px;

color#444;

font-size: 2.8em;

position: relative;

}

</style>

</head>

<body>

<section>

<div id="uno">1</div>

<div id="dos">2</div>

<div id="Tres">3</div>

</section>

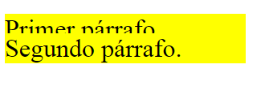
</body>

</html>

**Ejercicios**

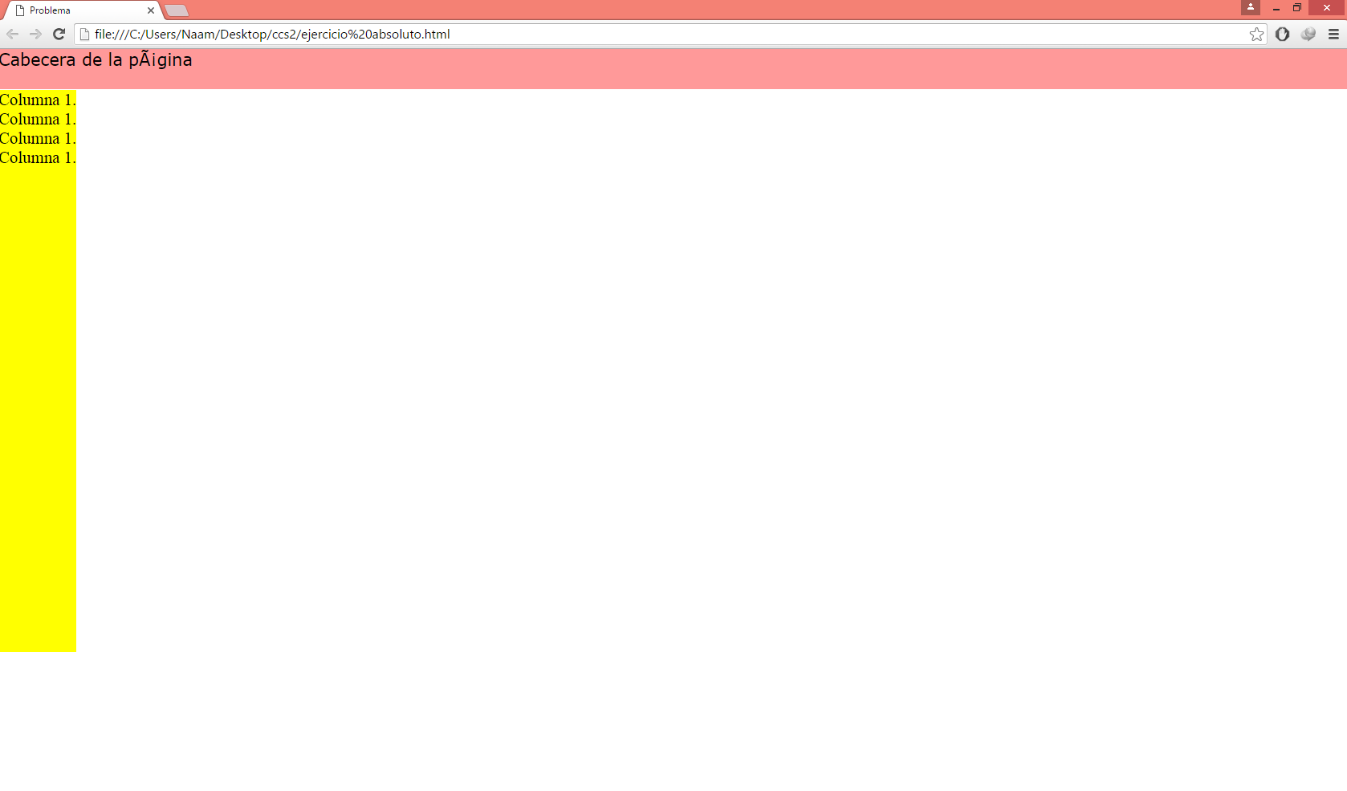
**Problema: posicionamiento relativo**

Disponer dos párrafos, desplazar el segundo párrafo hasta que quede casi superpuesto al primer párrafo.



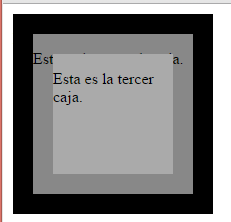
**Problema:**

Disponer dos div con coordenadas absolutas, uno que represente la cabecera de la página (50 píxeles de alto) y otro la primer columna de la izquierda.



**Problema:**

Mostrar 3 div con posiciones absolutas. Mostrarlos uno arriba de otro. Hacerlos de distintos tamaños. Mostrar más arriba el más pequeño y así sucesivamente



**Problema: posicionamiento fixed**

Crear una página que disponga dos div con posicionamiento fijo. Uno a la derecha y otro en la parte inferior de la página.



# Pseudo-elementos

Los selectores de CSS, las pseudo-clases y todos los elementos HTML no son suficientes para poder aplicar estilos a algunos elementos especiales. Si se desea por ejemplo cambiar el estilo de la primera línea de texto de un elemento, no es posible hacerlo con las utilidades anteriores.

La primera línea del texto normalmente es variable porque el usuario puede aumentar y disminuir la ventana del navegador, puede disponer de más o menos resolución en su monitor y también puede aumentar o disminuir el tamaño de letra del texto.

La única forma de poder seleccionar estos elementos especiales es mediante los pseudo-elementos definidos por CSS para este propósito.

### 3.5.1. El pseudo-elemento :first-line

El pseudo-elemento :first-line permite seleccionar la primera línea de texto de un elemento. Así, la siguiente regla CSS muestra en mayúsculas la primera línea de cada párrafo:

p:first-line { text-transform: uppercase; }

Este pseudo-elemento sólo se puede utilizar con los elementos de bloque y las celdas de datos de las tablas.

Se pueden combinar varios pseudo-elementos de tipo :first-line para crear efectos avanzados:

div:first-line { color: red; }

p:first-line { text-transform: uppercase; }

<div>

<p>Lorem ipsum dolor sit amet...</p>

<p>Lorem ipsum dolor sit amet...</p>

<p>Lorem ipsum dolor sit amet...</p>

</div>

En el ejemplo anterior, la primera línea del primer párrafo también es la primera línea del elemento<div>, por lo que se le aplican las dos reglas CSS y su texto se ve en mayúsculas y de color rojo.

### 3.5.2. El pseudo-elemento :first-letter

El pseudo-elemento :first-letter permite seleccionar la primera letra de la primera línea de texto de un elemento. De esta forma, la siguiente regla CSS muestra en mayúsculas la primera letra del texto de cada párrafo:

p:first-letter { text-transform: uppercase; }

Los signos de puntuación y los caracteres como las comillas que se encuentran antes y después de la primera letra también se ven afectados por este pseudo-elemento.

Este pseudo-elemento sólo se puede utilizar con los elementos de bloque y las celdas de datos de las tablas.

### 3.5.3. Los pseudo-elementos :before y :after

Los pseudo-elementos :before y :after se utilizan en combinación con la propiedad content de CSS para añadir contenidos antes o después del contenido original de un elemento.

Las siguientes reglas CSS añaden el texto Capítulo - delante de cada título de sección <h1> y el carácter . detrás de cada párrafo de la página:

h1:before { content: "Capítulo - "; }

p:after { content: "."; }

El contenido insertado mediante los pseudo-elementos :before y :after se tiene en cuenta en los otros pseudo-elementos :first-line y :first-letter.

**Ejemplo**

<!DOCTYPE html>

<html>

<head>

<title></title>

<style type="text/css">

.esfera{

width: 200px;

height: 200px;

border-radius: 50%;

background:-webkit-radial-gradient(top left, #00FFD4,#003F3D);

margin: auto;

position: relative;

}

.esfera:after{

content: "";

display: block;

width: 150px;

height: 5px;

background: black;

position: absolute;

bottom: 0px;

left: 50%;

border-radius: 50%;

box-shadow: 0 0 10px 10px #000;

opacity: 0.6;

z-index: -1;

}

</style>

</head>

<body>

<div class="esfera">

</div>

</body>

</html>

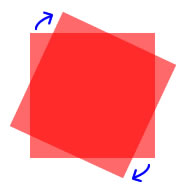
**Ejercicio**

**Transformaciones (scale, rotate, skew, traslate)**

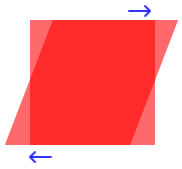
CSS3 ha incorporado un atributo muy útil que nos permite realizar diferentes transformaciones en nuestros elementos HTML. Dicho atributo es “**transform**”.

Con dicho atributo podemos realizar diferentes transformación como :

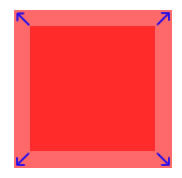
**Rotaciones (rotate)**

****

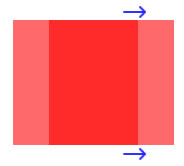
**Inclinados (skew)**



**Escalados (scale)**



**Desplazamientos (translate)**



El atributo “transform” debe ir acompañado por el valor que especifica el tipo de transformación a realizar (los anteriores), y además, uno o varios parámetros que configuran la transformación.  
Entonces el atributo completo tendría la siguiente forma:

[?](http://expertoencss3.com/curso/modulo-2/interface/transformaciones-scale-rotate-skew-traslate)

|  |
| --- |
| transform: tipo(parametros); |

He creado varios ejemplos para que veas los diferentes usos.

Primero configure un margen de documento de 30px (arriba y a la izquierda)  
Luego un caja DIV de 150px de ancho y alto, y color de fondo naranja.  
Y luego 3 clases CSS diferentes: una que rota al elemento, otra que lo inclina, otra que lo escala y otra que lo desplaza.  
Luego solo basta con aplicar cada clase a una caja o DIV diferente.

Código CSS que utilicé en los ejemplos.

**Rotar**

Aquí aplicamos una rotación de 25 grados.

[?](http://expertoencss3.com/curso/modulo-2/interface/transformaciones-scale-rotate-skew-traslate)

|  |
| --- |
| transform: rotate(25deg); |

**Inclinar**

Aquí aplicamos un inclinado vertical de 15 grados, y uno horizontal de 3 grados.

[?](http://expertoencss3.com/curso/modulo-2/interface/transformaciones-scale-rotate-skew-traslate)

|  |
| --- |
| transform: skew(15deg,3deg); |

**Escalar**

Aquí aplicamos un escalado horizontal del 120% (lo que equivale al valor 1.2), y uno vertical del 60% (lo que equivale a un valor de 0.6.  
Este valor funciona como un factor que se multiplica por la medida original, entonces si la medida original es de 150px y le aplicamos un valor de 1, dicho eje no se escalará, ya que 150 x 1 = 150.

[?](http://expertoencss3.com/curso/modulo-2/interface/transformaciones-scale-rotate-skew-traslate)

|  |
| --- |
| transform: scale(1.2,0.6); |

**Desplazar**

Aquí desplazamos el elemento 100px por el eje X, o sea, horizontalmente.  
Y 10 px hacia abajo, sobre el eje Y.

[?](http://expertoencss3.com/curso/modulo-2/interface/transformaciones-scale-rotate-skew-traslate)

|  |
| --- |
| transform: translate(100px, 19px); |

Recordemos que hay que utilizar los prefijos de los navegadores, para obtener compatibilidad. (-moz para Firefox y –webkit para Safari y Chrome)  
Esta funcionalidad no es compatible con IE7.

Este es el código completo del ejemplo:

[?](http://expertoencss3.com/curso/modulo-2/interface/transformaciones-scale-rotate-skew-traslate)

|  |
| --- |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "<http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd>">  <html xmlns="<http://www.w3.org/1999/xhtml>">  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Documento sin título</title>  <style type="text/css">  body {      margin-left: 30px;      margin-top: 30px;  }    #caja {      background-color: #F90;      height: 150px;      width: 150px;  }    .rotar {  transform: rotate(25deg);  -moz-transform: rotate(25deg);  -webkit-transform: rotate(25deg);  }    .inclinar {  transform: skew(15deg,3deg);  -moz-transform: skew(15deg,3deg);  -webkit-transform: skew(15deg,3deg);  }    .desplazar{  transform: translate(100px, 19px);  -moz-transform: translate(100px, 19px);  -webkit-transform: translate(100px, 19px);  }    .escalar {  transform: scale(1.2,0.6);  -moz-transform: scale(1.2,0.6);  -webkit-transform: scale(1.2,0.6);  }    </style>  </head>    <body>  <div id="caja">Caja sin transformación.</div><br />  <br />  <div id="caja">Caja rotada.</div>  <br />  <br />  <div id="caja">Caja inclinada.</div>  <br />  <br />  <div id="caja">Caja escalada.</div>  <br />  <br />  <div id="caja">Caja desplazada.</div>  </body>  </html> |

## Animaciones

Hasta el lanzamiento de CSS3, las animaciones web eran realizadas con tecnología Flash, JavaScript o Gifs animados.

Esta nueva versión de CSS nos permite desarrollar animaciones de una forma novedosa, con un código más limpio y fácil de escribir.

**Introducción**

Las animaciones dentro de una página web existen desde hace bastante tiempo y las mismas eran posibles gracias a diferentes tecnologías.  
Dejando de lado a las animaciones del tipo **GIF**, dentro de las que nos permiten buenos resultados y fluidez, podemos citar a **Flash**, que logró un gran avance en este campo, y a otras tecnologías como por ejemplo **Silverlight**.

Pero todas estas tecnologías que permitían la creación de animaciones, requerían que se instalaran componentes adicionales en los navegadores, ya que las mismas no estaban basadas en HTML o CSS estándar.

**NOTA**: existe otra tecnología que en este caso dejaremos de lado, que es el uso de las librerías JavaScript del tipo jQuery, con las que se pueden desarrollar animaciones interesantes.

Ahora bien, CSS3 incorpora una nueva tecnología para la creación de animaciones que es mucho mas simple que el uso de JavaScript o Flash.  
Además se esto, también posibilita la interacción con el usuario sin la necesidad de tener que desarrollar o escribir código.

**Ventajas**

Con las animaciones en CSS3 podemos prescindir en gran medida, del uso de Flash.  
Esto a simple vista deja entrever 3 grandes ventajas: la compatibilidad,  el posicionamiento y la facilidad de desarrollo.

Unos de los problemas mas grandes de compatibilidad entre dispositivos móviles, es el uso de Flash, ya que la mayoría no soporta esta tecnología. Con CSS3 nos olvidamos de este problema.

Por otro lado, el evitar el uso de Flash nos trae una segunda ventaja que es la de que nuestro sitio estará íntegramente desarrollado en HTML y CSS, con lo cual no tendremos problemas de posicionamiento en indexación en los buscadores como Google, Yahoo, MSN y otros.

Y por último, la facilidad de desarrollo, ya que no necesitaremos incursionar en otras tecnología para lograr un sitio visualmente atractivo.

**Desventajas**

Quizás hoy en dia, la mayor desventaja es el poco soporte que existe para esta tecnología.

Los diferentes navegadores no se han adatado 100% al uso de animaciones CSS, aunque van en camino a ello.

Por supuesto, el mayor problema de incompatibilidad lo encontramos en Internet Explorer.

Para comenzar, es necesario hacerlo por el principio:

* [**Conceptos Básicos para las animaciones CSS**](http://expertoencss3.com/curso/conceptos-basicos)

**Conceptos Básicos**

El uso de las animaciones en CSS3 enriquecerán notablemente cualquier sitio web.  
Pero para trabajar correctamente con ellas y sacar su máximo provecho, debemos conocer algunos conceptos básicos.

**Fotograma Clave (keyframe)**

Los fotogramas claves son, por lo general, el punto de comienzo y de fin de cada animación.  
Pero también se utilizan fotogramas clave o keyframes, en el transcurso de una animación para marcar algún cambio de estado de la misma.

Por ejemplo, para una animación con 5 keyframes a intervalos regulares, uno al principio, uno al final, y 4 intermedios, el código CSS se definiría de la siguiente forma:

[?](http://expertoencss3.com/curso/modulo-3/animaciones/conceptos-basicos)

|  |
| --- |
| @keyframes 'nombre\_fotograma\_clave' {  0% {  left: 50px;  }  25% {  left: 100px;  }  50% {  left: 75px;  }  75% {  left: 125px;  }  100% {  left: 100px;  }  } |

Como vemos, los keyframes se marcan con un porcentaje correspondiente al lugar de la animación donde se producirá el cambio o modificación de la misma (keyframe)  
Y dentro de los porcentajes (o keyframes) definimos los pixeles de longitud y la alineación donde se ubicaría el fotograma dentro del DIV contenedor.

Luego, para el DIV donde aplicaríamos la animación (en este caso llamado “caja”), utilizaríamos el siguiente código CSS:

[?](http://expertoencss3.com/curso/modulo-3/animaciones/conceptos-basicos)

|  |
| --- |
| caja {  animation-name: 'nombre-fotograma-clave';  animation-duration: 30s;  animation-iteration-count: 5;  } |

Lo que aquí definimos es:

**animation-name**: el nombre del fotograma clave (el mismo que le pusimos en @keyframes).  
**animation-duration**: la duración que tendrá la animación.  
**animation-iteration-count**: la veces que se repetirá la animación.

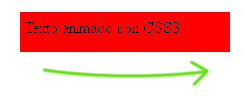
**NOTA**: esto es solo un ejemplo de la estructura de definición de una animación. No es funcional.

Recordemos que para esto, debemos utilizar los prefijos de los navegadores, sino no nos funcionará.  
Para eso debemos agregar antes de “@keyframes” y cada propiedad del DIV, el prefijo “-moz-“ (Firefox), o el prefijo “-webkit-“ (Chrome y Safari)

Tambien podemos aplicar otras propiedades de animación como:

**animation-delay**: indica el momento en el que comenzará la animación. Si es 0 se ejecuta cuando se carga la página.  
**animation-timing-function**: esta propiedad se aplica solo entre los fotogramas clave y especifica como progresa la animación a lo largo de un ciclo.  
**animation-direction**: define el sentido de la animación. Si ingresarmos “alternate” y los ciclos de iteración son impares, la animación irá en la dirección normal, si no se realizará en la dirección contraria.  
**animation**: esta propiedad combina las anteriores de una forma más resumida.

Veamos un ejemplo concreto de animación funcionando:

* **Texto Animado**
* 
* Ahora que ya hemos visto los conceptos básicos de animación CSS, vamos a crear un texto animado que se desplace de izquierda a derecha y vuelta al principio.
* Para eso, primero crearemos un DIV llamado “**caja**” con un texto negro dentro, fondo rojo, un padding de 5 px, posicionado relativamente para poder moverlo, con un alto de 30px y un ancho de 200px.
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/texto-animado)

|  |
| --- |
| width: 200px;  height: 30px;  background-color: #ff0000;  color: #000000;  position: relative;  padding: 5px; |

* Luego, también dentro de las propiedades del DIV, especificaremos los parámetros de animación para Chrome y Safari (-webkit) y para Firefox (-moz):
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/texto-animado)

|  |
| --- |
| -webkit-animation-name: movimiento-horizontal;  -webkit-animation-duration: 4s;  -webkit-animation-iteration-count: infinite;  -webkit-animation-direction: alternate;  -moz-animation-name: movimiento-horizontal;  -moz-animation-duration: 4s;  -moz-animation-iteration-count: infinite;  -moz-animation-direction: alternate; |

* Primero definimos el nombre de la animación como “**movimiento-horizontal**”.  
  Luego le dimos una duración de 4 segundos.  
  Seguido de eso le especificamos que se repita indefinidas veces.  
  En cuarto lugar le indicamos con “**alternate**”, que luego de llegar al  final de la animación, vuelva a su posición inicial realizando el camino inverso.  
  Las otras cuatro líneas son las mismas pero con el prefijo para Firefox (-moz)
* Luego debemos definir los keyframes, en este caso solo 2: el primero y el ultimo.
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/texto-animado)

|  |
| --- |
| @-webkit-keyframes movimiento-horizontal {  from {  left: 0px;  }  to {  left: 200px;  }  }  @-moz-keyframes movimiento-horizontal {  from {  left: 0px;  }  to {  left: 200px;  }  } |

* Aquí le indicamos que nuestro div se moverá desde la posición 0px alineado contra la izquierda, hasta la posición 200px también alineado contra la izquierda. (también para Firefox y Chrome)
* Luego, en el cuerpo del documento HTML, creamos un DIV llamado “**caja**” con un texto de ejemplo:
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/texto-animado)

|  |
| --- |
| <div id="caja">Texto animado con CSS3.</div> |

* Y listo, tenemos creado nuestro texto animado en CSS de forma muy sencilla y simple.
* El código completo de la pagina seria:
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/texto-animado)

|  |
| --- |
| <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  <title>Texto Animado en CSS3</title>  <style type="text/css">  #caja {  -webkit-animation-name: movimiento-horizontal;  -webkit-animation-duration: 4s;  -webkit-animation-iteration-count: infinite;  -webkit-animation-direction: alternate;/\*vuelve a su posicion inicial \*/  -moz-animation-name: movimiento-horizontal;  -moz-animation-duration: 4s;  -moz-animation-iteration-count: infinite;  -moz-animation-direction: alternate;/\*vuelve a su posicion inicial \*/    width: 200px;  height: 30px;  background-color: #ff0000;  color: #000000;  position: relative;  padding: 5px;  }  @-webkit-keyframes movimiento-horizontal {  from {  left: 0px;  }  to {  left: 200px;  }  }    @-moz-keyframes movimiento-horizontal {  from {  left: 0px;  }  to {  left: 200px;  }  }  </style>  </head>    <body>  <div id="caja">Texto animado con CSS3.</div>  </body>  </html> |

**Ejercicio**

<!DOCTYPE html>

<html>

<head>

<title>Animación</title>

<style type="text/css">

div{

width: 100px;

height: 100px;

background: blue;

position: relative;

animation-name:movimiento;

animation-duration:10s;

}

@keyframes movimiento{

0%{

background: blue;

left:0px;

top:0px;

}

20%{

background: red;

left:100px;

top:0px;

}

50%{

background: yellow;

left:100px;

top:100px;

}

75%{

background: blue;

left:0px;

top:100px;

}

100%{

background: blue;

left:0px;

top:0px;

}

}

</style>

</head>

<body>

<div>

</div>

</body>

</html>

**Transiciones**

* Las transiciones de CSS 3 han abierto un abanico de nuevas posibilidades de interactividad.
* Una transición es algo que ocurre entre un momento y otro.
* Si bien el típico efecto de hover, es decir, el que al pasar el mouse sobre un elemento este cambio, es una transición en si. Pero pasa tan rápido que prácticamente no nos damos cuenta que sucede.  
  Por otro lado, tampoco podemos controlar el tiempo que eso tarda.
* Las nuevas propiedades de CSS3,  nos permiten controlar el tiempo que durará ese cambio entre un estado y otro lo que generará una animación simple.
* Vamos a ver un ejemplo donde tenemos una caja o DIV rectangular, y al pasar por encima con el mouse, esta cambia de estado fundiéndose de un color a otro.
* Puedes ver el ejemplo aquí:
* [http://expertoencss3.com/curso/wp-content/uploads/2011/12/bot_demo.png](http://expertoencss3.com/curso/ejemplos/transiciones.html)
* La propiedad para logar esto se llama “**transition** “ y debemos hacer uso de los  prefijos de cada navegador:
* **-moz-transition** (Firefox), **-webkit-transition** (Chrome y Safari), **-o-transition** (Opera).
* En realidad esta propiedad está en su formato resumido, y por lo general, no suele usarse asi,  sino que se “divide” del mismo modo que lo hacemos con muchas otras.
* En este caso, son cuatro:
* **transition-property**  
  **transition-duration**  
  **transition-timing-function**  
  **transition-delay**
* Aquí la explicación de cada una:
* **transition-property**  
  Esta indica cual o cuales propiedades del elemento vamos a modificar; si son varias, las separamos con comas:
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/transiciones)

|  |
| --- |
| transition-property: color;  transition-property: color, opacity, width; |

* **transition-timing-function** (no utilizada en este ejemplo)Especifica como transcurrirá el tiempo de la animacion:
* **linear** - La animación se realiza de manera uniforme.  
  **ease** – La animación acelera al inicio se retarda un poco y se acelera al final de nuevo.  
  **ease-in** - La animación se retarda al inicio, pero lo repone al final.  
  **ease-out** – La animación se acelera al inicio pero se retarda al final.  
  **ease-in-out** - La animación se retarda al inicio se acelera un poco luego se retarda al final de nuevo.
* **transition-duration**  
  Aquí indicamos el tiempo que durará el efecto. Podemos especificar uno y varios valores separados por comas; si lo hacemos, estamos diciendo al navegador que cada propiedad indicada en  transition-property tiene un tiempo de transicion distinto.
* El valor está expresado en segundos:
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/transiciones)

|  |
| --- |
| transition-duration: 1s;  transition-duration: 1s, 3s; |

* Entonces para nuestro ejemplo, primero creamos un DIV y le aplicamos una clase llamada “demo” que luego crearemos:
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/transiciones)

|  |
| --- |
| <div class="demo"></div> |

* Ahora creamos la clase “demo” en sus 2 estados: el normal y el correspondiente al hover, es decir, cuando pasamos el mouse por encima:
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/transiciones)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | .demo {   width:200px;   height:30px;   background-color:#FFF;   cursor: pointer;   border: 4px solid #78999C;   -moz-transition-property: background-color;   -moz-transition-duration: 2s;   -webkit-transition-property: background-color;   -webkit-transition-duration: 2s;   -o-transition-property: background-color;   -o-transition-duration: 2s;   }   .demo:hover {   background-color:#6A2B00;   -moz-transition-property: background-color;   -moz-transition-duration: 2s;   -webkit-transition-property: background-color;   -webkit-transition-duration: 2s;   -o-transition-property: background-color;   -o-transition-duration: 2s;   } |

* Lo que indicamos aquí es que el DIV tenga un ancho de 200px, un alto de 30px, color de fondo blanco (#FFF), que el cursor encima del sea de tipo “ponter” (flechita), un borde de 4px color #78999C y luego la especificación de la transición:
* **transition-property: background-color;** (especificando que lo que voy a cambiar es el color de fondo)  
  **transition-duration: 2s;** (indicando que el tiempo será de 2 segundos)
* Luego de esto, configuramos las propiedades para cuando pasemos por encima con el mouse y lo único que modificamos es el color de fondo a #6A2B00, y luego la misma transición para que vuelva al color original.
* El código final completo seria:
* [?](http://expertoencss3.com/curso/modulo-3/animaciones/transiciones)

|  |
| --- |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "<http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd>">   <html xmlns="<http://www.w3.org/1999/xhtml>">   <head>   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />   <title>Transiciones en CSS3</title>   <style type="text/css">   .demo {   width:200px;   height:30px;   background-color:#FFF;   cursor: pointer;   border: 4px solid #78999C;   -moz-transition-property: background-color;   -moz-transition-duration: 2s;   -webkit-transition-property: background-color;   -webkit-transition-duration: 2s;   -o-transition-property: background-color;   -o-transition-duration: 2s;   }   .demo:hover {   background-color:#6A2B00;   -moz-transition-property: background-color;   -moz-transition-duration: 2s;   -webkit-transition-property: background-color;   -webkit-transition-duration: 2s;   -o-transition-property: background-color;   -o-transition-duration: 2s;   }   body {   background-color: #0C2129;   color: #FFF;   }   </style>   </head>  <body>   <h1>Transiciones CSS3</h1>   <h3>  Pasa el mouse sobre el rectángulo   </h3>   <div></div>   </body>   </html> |