CSS3

CSS3 - Selector por atributos

- Ahora podemos referenciar un elemento no solo por los atributos **id** y **class** sino también a través de cualquier otro atributo:

```
p[name] { font-size: 20px }
```

Con esto damos 20px de tamaño de fuente a todos los <p> con el atributo name.

- También podemos especificar el valor del atributo:
  - `p[name="mitexto"] { font-size: 20px }`

Con esto damos 20px de tamaño de fuente a todos los <p> con el atributo name y valor "mitexto"

CSS3 - Selector por atributos

- CSS3 permite combinar "**=**" con otros operadores para hacer una selección más específica (Puedes usar expresiones regulares).

```
p[name^="mi"] { font-size: 20px }
p[name$="mi"] { font-size: 20px }
p[name*="mi"] { font-size: 20px }
```

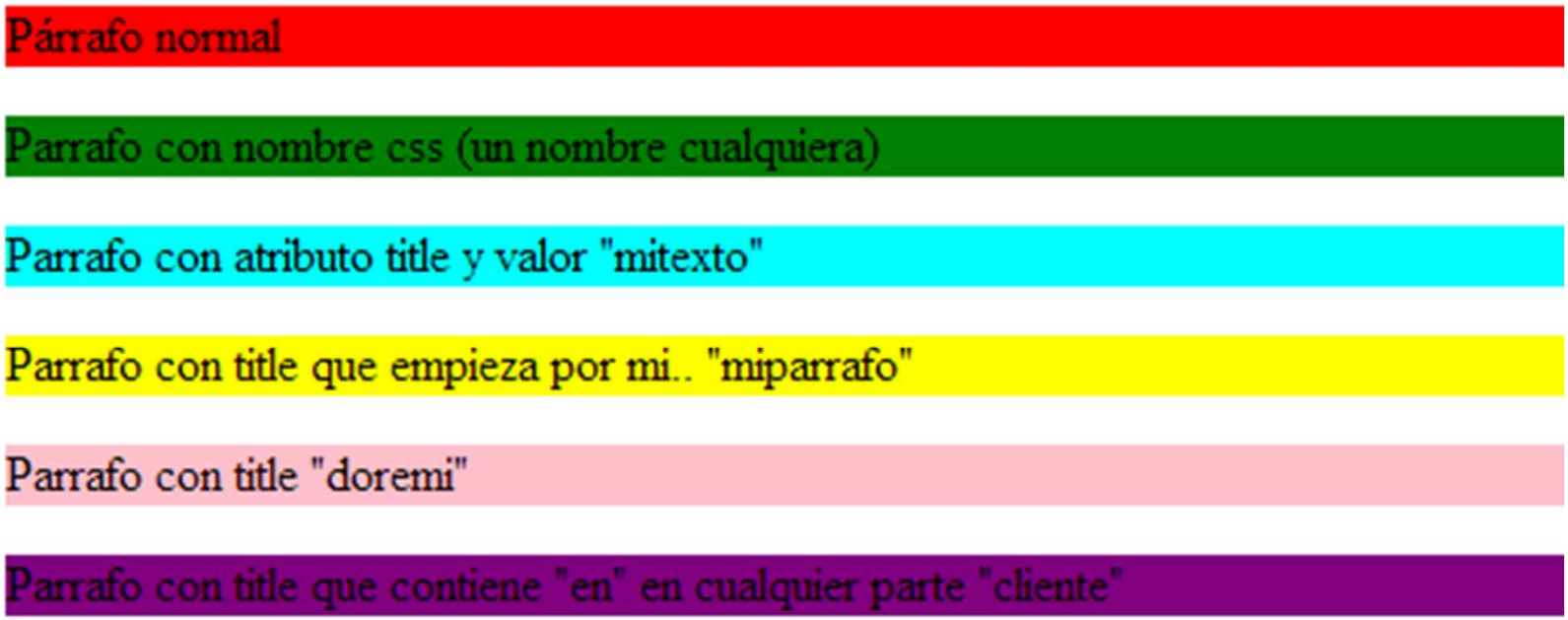
- La regla con el selector **^=** será asignada a todo elemento **<p>** que contiene un atributo **name** con un valor comenzado en "**mi**" (por ejemplo, "**mitexto**", "**micasa**").
- La regla con el selector **$=** será asignada a todo elemento **<p>** que contiene un atributo **name** con un valor finalizado en "**mi**" (por ejemplo "**textomi**", "**casami**").
- La regla con el selector ***=** será asignada a todo elemento **<p>** que contiene un atributo **name** con un valor que incluye el texto "**mi**" (en este caso, el texto podría también encontrarse en el medio, como en "**textomicasa**").

Ejemplo. CSS y HTML

```
<html>
<head>
    <title>Ejemplo8</title>
    <style>
        p{ background-color: red; }
        p[name] { background-color: green; }
        p[title="texto"] { background-color: cyan; }
        p[title^="mi"] { background-color: yellow; }
        p[title$="mi"] { background-color: pink; }
        p[title*="en"] { background-color: purple; }
    </style>
</head>
<body>
    <p>Párrafo normal</p>
    <p name="css">Parrafo con nombre css (un nombre cualquiera)</p>
    <p title="texto">Parrafo con atributo title y valor "mitexto"</p>
    <p title="miparrafo">Parrafo con title que empieza por mi.. "miparrafo"</p>
    <p title="doremi">Parrafo con title "doremi"</p>
    <p title="cliente">Parrafo con title que contiene "en" en cualquier parte "cliente"</p>
</body>
</html>
```

(Ver CSS 3 - Ejemplo 1.html)

Ejemplo. Demostración



Referenciando con pseudo-clases

- Usando pseudo clases podemos aprovechar la organización de los elementos en el código y referenciar un elemento específico sin importar cuánto conocemos sobre sus atributos y el valor de los mismos.
- Los selectores de pseudo clases son:
  - `nth-child(n)`
  - `first-child`
  - `last-child`
  - `only-child`
  - `not`

nth-child(n)

- nth-child(n) permite seleccionar el elemento que queramos indicando el índice (siendo el índice la posición tomando la lista de todos los elementos con ese selector)
- Supongamos que tenemos algo así:

```
<p class="mitexto1">Mi texto1</p>
<p class="mitexto2">Mi texto2</p>
<p class="mitexto3">Mi texto3</p>
<p class="mitexto4">Mi texto4</p>
```

Añadiendo en el css *p:nth-child(2){background:blue}* cambiaremos el fondo a el segundo elemento p (mitexto2) por su posición, sin usar atributos, clases o id's. Hay 2 comodines usados en páginas dinámicas para usar como índices (*odd* y *even*)

Ejemplo. nth-child(n)

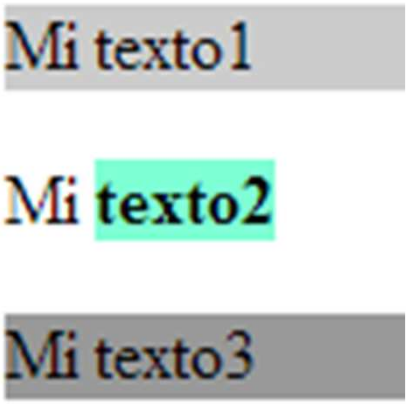
```
<style>
      p:nth-child(1){ background: green; }
      p:nth-child(2){ background: red; }
      p:nth-child(3){ background: cyan; }
      p:nth-child(4){ background: pink; }
      tr:nth-child(odd){ background: #CCC; }
      tr:nth-child(even){ background: blue; }
</style>
```

(Ver CSS 3 - Ejemplo 2.html)

First-child,last-child,only-child

- Existen otras importantes pseudo clases relacionadas con esta última, como **first-child**, **last-child** y **only-child**.
  - La pseudo clase **first-child** referencia solo el primer hijo.
  - **last-child** referencia solo el último hijo.
  - **only-child** afecta un elemento siempre y cuando sea el único

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Ejemplo 8-2</title>
    <style>
        p:first-child{ background:#CCC; }
        p:last-child{ background:#999; }
        b:only-child{ background:aquamarine; }
    </style>
</head>
<body>
    <div id="wrapper">
        <p class="mitexto1">Mi texto1</p>
        <p class="mitexto2">Mi <b>texto2</b></p>
        <p class="mitexto3">Mi texto3</p>
    </div>
</body>
```



(Ver CSS 3 - Ejemplo 3.html)

Pseudo clase not()

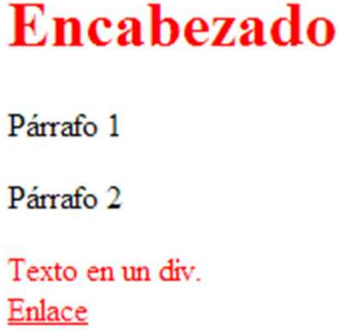
- Otra importante pseudo clase llamada **not()** es utilizada para realizar una negación:

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

La regla asignará un margen de 0 pixeles a cada elemento del documento excepto los elementos **<p>**.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Ejemplo 8-3</title>
    <style>
        p { color:#000000; }
        :not(p){ color:red }
    </style>
</head>
<body>
    <h1>Encabezado</h1>
    <p>Párrafo 1</p>
    <p>Párrafo 2</p>
    <div>Texto en un div.</div>
    <a href="http://www.google.com" target="_blank">Enlace</a>

</body>
</html>
```



(Ver CSS 3 - Ejemplo 4.html)

Selector de hijos >

- Se utiliza para seleccionar un elemento que es hijo directo de otro elemento.
- En el ejemplo el selector de hijos obliga a que el elemento <b> sea hijo directo de un elemento <p>. Por lo tanto, el estilo del selector p > b no se aplican al segundo texto.

```
<style type="text/css">
        p > b { color: red; }
</style> </head>
<body>
        <p><b>Texto 1</b></p>
        <p><span><b>Texto 2</b></span></p>
</body>
```

(Ver CSS 3 - Ejemplo 5.xhtml)

Selector adyacente +

- El elemento a la derecha del signo + debe cumplir las dos siguientes condiciones:
  - El elemento a la izquierda y a la derecha deben ser hermanos, por lo que su elemento padre debe ser el mismo.
  - El elemento a la derecha debe aparecer inmediatamente después del elemento a la izquierda en el código HTML de la página.

Ejemplo. Selector adyacente +

```
<head>
     <style type="text/css">
     <!-- SELECTOR ADYACENTE: AFECTA SOLO AL SUBTITULO1 -->
      h1 + h2 { color: red; }
     </style>
</head>
<body>
     <div>
          <h1>Titulo1</h1> <!-- TITULO1-->
          <h2>Subtitulo1</h2> <!-- SUBTITULO1-->
          <h2>Subtitulo2</h2> <!-- SUBTITULO2-->
     </div>
     <h2>Subtitulo3</h2> <!-- SUBTITULO3-->
     <h1>Titulo2</h1> <!-- TITULO2-->
</body>
```

(Ver CSS3 - Ejemplo 6.xhtml)

Selector de referencia ~

- Este es parecido al anterior, pero no se limita a solo al primer elemento sino que selecciona todos los elementos que son hermanos del primero, en este caso selecciona todos los elementos h2 que son hermanos al h1.

```
<head>
    <style type="text/css">
        h1 ~ h2 { color: red; }
    </style>
</head>
<body>
    <div>
        <h1>Titulo1</h1>  <!-- TITULO1-->
        <h2>Subtitulo1</h2> <!-- SUBTITULO1-->
        <h2>Subtitulo2</h2> <!-- SUBTITULO2-->
    </div>
        <h2>Subtitulo3</h2> <!-- SUBTITULO3-->
 </body>
```

(Ver CSS3 - Ejemplo 7.xhtml)

border-radius

- Con esta propiedad podemos definir bordes redondeados. Con la propiedad border-radius también se definen las propiedades border-top-right-radius, border-bottom-right-radius, border-bottom-left-radius, border-top-left-radius para establecer la curvatura de cada esquina.
- Para que el borde sea visible debemos definir también la propiedad border. Con la propiedad border también se definen las propiedades border-top, border-bottom, border-left, border-right.

Ejemplo. border-radius

```
<head>
  <style type="text/css">
       .borde{
          border:2px solid black; <!-- ANCHURA TIPO COLOR -->
          -webkit-border-radius: 20px; <!-- WEBKIT -->
          -moz-border-radius: 20px; <!-- GECKO -->
          border-radius: 20px; <!-- CSS3 GENERICO -->
          width:200px; height:200px;
       }
  </style>
</head>
<body>
       <div class="borde">
       </div>
</body>
```

(Ver CSS3 - Ejemplo 8.xhtml)

box-shadow

- Con esta propiedad podemos definir sombra en las cajas. Tiene varios parámetros:
  - La primera medida es obligatoria e indica el desplazamiento horizontal de la sombra. Si el valor es positivo, la sombra se desplaza hacia la derecha y si es negativo, se desplaza hacia la izquierda.
  - La segunda medida también es obligatoria e indica el desplazamiento vertical de la sombra. Si el valor es positivo, la sombra se desplaza hacia abajo y si es negativo, se desplaza hacia arriba.

box-shadow

- La tercera medida es opcional e indica el radio utilizado para difuminar la sombra. Cuanto más grande sea su valor, más borrosa aparece la sombra. Si se utiliza el valor 0, la sombra se muestra como un color sólido.
- La cuarta medida también es opcional e indica el radio con el que se expande la sombra. Si se establece un valor positivo, la sombra se expande en todas direcciones. Si se utiliza un valor negativo, la sombra se comprime.
- El color indicado es directamente el color de la sombra que se muestra.

Ejemplo. box-shadow

```
<head>
  <style type="text/css">
       .sombra{
          -webkit-box-shadow: 2px 2px 5px #999;
          -moz-box-shadow: 2px 2px 5px #999;
           box-shadow: 2px 2px 5px #999;
          width:200px; height:200px;
       }
  </style>
</head>
<body>
       <div class=“sombra">
       </div>
</body>
```

(Ver CSS3 - Ejemplo 9.xhtml)

CSS3 - text-shadow

Sirve para sombrear un texto.

Tiene cuatro parámetros, dos de ellos obligatorios:

- Desplazamiento horizontal de la sombra respecto al texto (x-offset).
- Desplazamiento vertical (y-offset).

Los parámetros opcionales indican lo nítido o borroso (blur) que se ve la sombra y el color de la misma.

```
h1 {text-shadow: 3px 3px 2px red;}
```

Ejemplo. text-shadow

```
<html>
  <head>
      <title>text-shadow</title>
      <style>
           h1{text-shadow: 5px 4px #ED1C24;}
           h2{text-shadow: 1px 2px 2px #0071BC;}
      </style>
  </head>
  <body>
      <h1>EJEMPLO DE TEXT-SHADOW</h1>
      <h2>OTRO EJEMPLO DE TEXT-SHADOW</h2>
  </body>
</html>
```

(Ver CSS3 - Ejemplo 10.html)

CSS3 - @font-face

Permite definir las fuentes que utiliza una página web.

Se usa cuando no son tipografías estándar.

En la descripción puede indicarse la URL desde la que el navegador se puede descargar la fuente utilizada si el usuario no dispone de ella.

También permite definir otras propiedades de la fuente, como su formato, grosor y estilo.

```
@font-face {
        font-family: Vivaldi;
        src: url(VIVALDI0.eot);
}
```

(Ver CSS3 - Ejemplo 11.html)

CSS3 - linear-gradient

Esta propiedad se le asigna a un atributo background y sirve para crear un degradado lineal que va desde un color a otro. Puede ser de arriba a abajo, de izquierda a derecha y viceversa.

Pueden hacerse "color stops", que consiste en declarar el lugar desde donde debe empezar el gradiente del color.

```
degradado {
    background: -webkit-linear-gradient(top left, #fff, #f66);
    background: -moz-linear-gradient(top right, #fff, #f66);
    background: -o-linear-gradient(45º, #fff, #f66);
    background: linear-gradient(#fff 50%, #f66);
 }
```

CSS3 - radial-gradient

Sirve para crear degradados formando diseños circulares o elípticos.

Parámetros:

-Posición inicial del gradiente circular: especificado con coordenadas. Si se omite, empieza en el centro del fondo del elemento.

-Forma : circle o ellipse.

-Colores del degradado y paradas de color.

```
degradado{
    background: -webkit-radial-gradient(circle, #66f, #ffc);
    background: -moz-radial-gradient(circle, #66f, #ffc);
    background: radial-gradient(circle, #66f, #f80, #ffc);
}
```

(Ver CSS3 - Ejemplo 12.html)

CSS3 - RGBA

Es una manera de especificar colores en la que se definen cuatro valores. RGB corresponde a rojo, verde y azul y el cuarto parámetro es el canal Alpha, que es el grado de transparencia del color.

Este canal tiene un valor entre cero y uno, siendo 0 totalmente transparente y 1 totalmente opaco.

```
div {background-color: rgba(303, 0, 100, 0.1);}
div2 {background-color: rgba(255, 125, 0, 0.7);}
div3 {background-color: rgba(146, 157, 0, 1);}
```

(Ver CSS3 - Ejemplo 13.html)

CSS3 - HSLA

Es otro modelo de definición de colores: Hue (tono), Saturation, Ligthness y el canal Alpha, con el mismo cometido que en RGBA.

Hue puede tomar valores entre 0 y 360.

(0 – rojo , 120 – verde, 240 – azul, 360- rojo)

Saturation y Lightness se expresan en porcentajes.

```
div {background-color: hsla(135, 60 %, 20%, 0.5);}
```

(Ver CSS3 - Ejemplo 14.html)

CSS3 - outline

Se utiliza para establecer de forma abreviada el valor de una o más propiedades. Se parece a la propiedad border pero con algunas diferencias: los perfiles no ocupan espacio y pueden tener formas no rectangulares.

Los valores permitidos son:

-outline-color: color | invert | inherit.

-outline-style: none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset | inherit.

-outline-width: media | thin | medium | thick | inherit.

```
div { outline: 5px solid #369; }
```

(Ver CSS3 - Ejemplo 15.html)

CSS3 - border-image

Con este atributo podemos utilizar una imagen como borde de un elemento de la página.

Otros atributos relacionados son:

border-image-source

border-image-slice

border-image-width

border-image-repeat

```
div {
   -moz-border-image: url(borde.png) 2 2 2 2;
   -webkit-border-image: url(borde.png) 2 2 2 2;
}
```

(Ver CSS3 - Ejemplo 16.html)

CSS 3 – transform y transition

- Los elementos HTML, cuando son creados, son como bloques sólidos e inamovibles. Pueden ser movidos usando código Javascript o aprovechando librerías populares como jQuery (*www.jquery.com*), por ejemplo, pero no existía un procedimiento estándar para este propósito hasta que CSS3 presentó las propiedades **transform** y **transition**.
- Ahora ya no tenemos que pensar en cómo hacerlo. En su lugar, solo tenemos que conocer cómo ajustar unos pocos parámetros y nuestro sitio web puede ser tan flexible y dinámico como lo imaginamos.
- La propiedad **transform** puede operar cuatro transformaciones básicas en un elemento: **scale** (escalar), **rotate** (rotar), **skew** (inclinar) y **translate** (trasladar o mover).

```
-ms-transform = Internet Explorer
-webkit-transform = Google Chrome
-moz-transform = Mozilla Firefox
-o-transform = Opera
```

CSS 3 – transform: scale

- La función **scale** recibe dos parámetros: el valor **X** para la escala horizontal y el valor **Y** para la escala vertical. Si solo un valor es provisto el mismo valor es aplicado a ambos parámetros.
- Números enteros y decimales pueden ser declarados para la escala. Esta escala es calculada por medio de una matriz. Los valores entre 0 y 1 reducirán el elemento, un valor de 1 mantendrá las proporciones originales y valores mayores que 1 aumentarán las dimensiones del elemento de manera incremental.
- Un efecto atractivo puede ser logrado con esta función otorgando valores negativos. Se invertiría la imagen.
- Existen también otras dos funciones similares a **scale** pero restringidas a la dimensión horizontal o vertical: **scaleX** y **scaleY**.

```
Elemento { transform: scale(valorx ,valory); }
```

CSS 3 – transform: rotate

- La función **rotate** rota el elemento en la dirección de las agujas de un reloj. El valor debe ser especificado en grados usando la unidad "**deg**".
- Si un valor negativo es declarado, solo cambiará la dirección en la cual el elemento es rotado.

```
-transform: rotate(30deg);
-transform: rotate(-30deg);
```

CSS3 – transform: skew

- Esta función cambia la simetría del elemento en grados y en ambas dimensiones.
- La función **skew** usa dos parámetros, pero a diferencia de otras funciones, cada parámetro de esta función solo afecta una dimensión (los parámetros actúan de forma independiente).Si solo declaramos el primer parámetro, solo la dimensión horizontal de la caja será modificada. Si usáramos los dos parámetros, podríamos alterar ambas dimensiones del objeto. Como alternativa podemos utilizar funciones diferentes para cada una de ellas: **skewX** y **skewY**.

```
-transform: skew(20deg);
```

CSS3 – transform: translate

- Similar a las viejas propiedades **top** y **left**, la función **translate** mueve o desplaza el elemento en la pantalla a una nueva posición.
- La función **translate** considera la pantalla como una grilla de pixeles, con la posición original del elemento usada como un punto de referencia. La esquina superior izquierda del elemento es la posición **0,0**, por lo que valores negativos moverán al objeto hacia la izquierda o hacia arriba de la posición original, y valores positivos lo harán hacia la derecha o hacia abajo.
- Dos valores pueden ser declarados en esta función si queremos mover el elemento horizontal y verticalmente, o podemos usar funciones independientes llamadas **translateX** y **translateY**.

```
-transform: translate(100px);
```

CSS3 – Transformación dinámica

- Lo que hemos aprendido hasta el momento en este capítulo cambiará la forma de la web, pero la mantendrá tan estática como siempre. Sin embargo, podemos aprovecharnos de la combinación de transformaciones y pseudo clases para convertir nuestra página en una aplicación dinámica.

```
#id:hover{
     transform: rotate(5deg);
}
```

(Ver CSS3 - Ejemplo 17.html)

CSS3 – Transition

- Una animación real requiere de un proceso de más de dos pasos.
- La propiedad **transition** fue incluida para suavizar los cambios, creando mágicamente el resto de los pasos que se encuentran implícitos en el movimiento. Solo agregando esta propiedad forzamos al navegador a tomar cartas en el asunto, crear para nosotros todos esos pasos invisibles, y generar una transición suave desde un estado al otro.

```
-moz-transition: -moz-transform 1s ease-in-out 0.5s;
```

  - El primer valor es la propiedad que será considerada para hacer la transición.
  - El segundo parámetro especifica el tiempo que la transición se tomará para ir de la posición inicial a la final.
  - El tercer parámetro puede ser cualquiera de las siguientes palabras clave: **ease**, **linear**, **ease-in**, **ease-out** o **ease-in-out**.
  - El último parámetro para la propiedad **transition** es el retardo. Éste indica cuánto tiempo tardará la transición en comenzar.

CSS3 – Transition

- Modo de transition:
  - Ease: Un poco más rápido al principio que al final
  - Linear: Velocidad estable
  - ease-in: Lento al principio, rápido al final.
  - ease-out: Rápido al principio, lento al final.
  - ease-in-out: Lento al principio y al final
  - cubic-bezier: super rápido, casi sin transición.

(Ver CSS3 - Ejemplo 18.html)