Referenciando con cualquier atributo

Aunque los métodos de referencia estudiados anteriormente cubren un variado espectro de situaciones, a veces no son suficientes para encontrar el elemento exacto. La última versión de CSS ha incorporado nuevas formas de referenciar elementos HTML. Uno de ellas es el **Selector de Atributo**. 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 }
```

Listado 2-11. *Referenciando solo elementos <p> que tienen el atributo name.*

La regla en el Listado 2-11 cambia solo elementos **<p>** que tienen un atributo llamado **name**. Para imitar lo que hicimos previamente con los atributos **id** y **class**, podemos también especificar el valor del atributo:

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

Listado 2-12. *Referenciando elementos <p> que tienen un atributo name con el valor mitexto.*

CSS3 permite combinar “**=**” con otros para hacer una selección más específica:

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

Listado 2-13. *Nuevos selectores en CSS3.*

Si usted conoce **Expresiones Regulares** desde otros lenguajes como Javascript o PHP, podrá reconocer los selectores utilizados en el Listado 2-13. En CSS3 estos selectores producen similares resultados:

- 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**”).

En estos ejemplos usamos el elemento **<p>**, el atributo **name**, y una cadena de texto al azar como “**mi**”, pero la misma técnica puede ser utilizada con cualquier atributo y valor que necesitemos. Solo tiene que escribir los corchetes e insertar entre ellos el nombre del atributo y el valor que necesita para referenciar el elemento HTML correcto.

Referenciando con pseudo clases

CSS3 también incorpora nuevas pseudo clases que hacen la selección aún más específica.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <title>Este texto es el título del documento</title>
  <link rel="stylesheet" href="misestilos.css">
</head>
<body>
  <div id="wrapper">
    <p class="mitexto1">Mi texto1</p>
    <p class="mitexto2">Mi texto2</p>
    <p class="mitexto3">Mi texto3</p>
    <p class="mitexto4">Mi texto4</p>
  </div>
```

```
</body>
</html>
```

Listado 2-14. *Plantilla para probar pseudo clases.*

Miremos por un momento el nuevo código HTML del Listado 2-14. Contiene cuatro elementos **<p>** que, considerando la estructura HTML, son hermanos entre sí e hijos del mismo elemento ** **.

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

```
p:nth-child(2){
  background: #999999;
}
```

Listado 2-15. *Pseudo clase nth-child().*

La pseudo clase es agregada usando dos puntos luego de la referencia y antes del su nombre. En la regla del Listado 2-15 referenciamos solo elementos **<p>**. Esta regla puede incluir otras referencias. Por ejemplo, podríamos escribirla como **.miclase:nth-child(2)** para referenciar todo elemento que es hijo de otro elemento y tiene el valor de su atributo **class** igual a **miclase**. La pseudo clase puede ser aplicada a cualquier tipo de referencia estudiada previamente.

La pseudo clase **nth-child()** nos permite encontrar un hijo específico. Como ya explicamos, el documento HTML del Listado 2-14 tiene cuatro elementos **<p>** que son hermanos. Esto significa que todos ellos tienen el mismo padre que es el elemento ** **. Lo que esta pseudo clase está realmente indicando es algo como: "el hijo en la posición…" por lo que el número entre paréntesis será el número de la posición del hijo, o índice. La regla del Listado 2-15 está referenciando cada segundo elemento **<p>** encontrado en el documento.

> **Hágalo usted mismo:** Reemplace el código en su archivo HTML por el del Listado 2-14 y abra el archivo en su navegador. Incorpore las reglas estudiadas en el Listado 2-15 dentro del archivo **misestilos.css** para comprobar su funcionamiento.

Usando este método de referencia podemos, por supuesto, seleccionar cualquier hijo que necesitemos cambiando el número de índice. Por ejemplo, la siguiente regla tendrá impacto sólo sobre el último elemento **<p>** de nuestra plantilla:

```
p:nth-child(4){
  background: #999999;
}
```

Listado 2-16. *Pseudo clase nth-child().*

Como seguramente se habrá dado cuenta, es posible asignar estilos a todos los elementos creando una regla para cada uno de ellos:

```
*{
  margin: 0px;
}

p:nth-child(1){
  background: #999999;
}
p:nth-child(2){
  background: #CCCCCC;
}
p:nth-child(3){
  background: #999999;
}
p:nth-child(4){
  background: #CCCCCC;
}
```

Listado 2-17. *Creando una lista con la pseudo clase nth-child().*

La primera regla del Listado 2-17 usa el selector universal ***** para asignar el mismo estilo a cada elemento del documento. Este nuevo selector representa cada uno de los elementos en el cuerpo del documento y es útil cuando necesitamos establecer ciertas reglas básicas. En este caso, configuramos el margen de todos los elementos en 0 pixeles para evitar espacios en blanco o líneas vacías como las creadas por el elemento **<p>** por defecto.

En el resto del código del Listado 2-17 usamos la pseudo clase **nth-child()** para generar un menú o lista de opciones que son diferenciadas claramente en la pantalla por

Hágalo usted mismo: Copie el último código dentro del archivo CSS y abra el documento HTML en su navegador para comprobar el efecto.

Para agregar más opciones al menú, podemos incorporar nuevos elementos **<p>** en el código HTML y nuevas reglas con la pseudo clase **nth-child()** usando el número de índice adecuado. Sin embargo, esta aproximación genera mucho código y resulta imposible de aplicar en sitios webs con contenido dinámico. Una alternativa para obtener el mismo resultado es aprovechar las palabras clave **odd** y **even** disponibles para esta pseudo clase:

```
*{
  margin: 0px;
}
p:nth-child(odd){
  background: #999999;
}
p:nth-child(even){
  background: #CCCCCC;
}
```

Listado 2-18. *Aprovechando las palabras clave odd y even.*

Ahora solo necesitamos dos reglas para crear la lista completa. Incluso si más adelante agregamos otras opciones, los estilos serán asignados automáticamente a cada una de ellas de acuerdo a su posición. La palabra clave **odd** para la pseudo clase **nth-child()** afecta los elementos **<p>** que son hijos de otro elemento y tienen un índice impar. La palabra clave **even**, por otro lado, afecta a aquellos que tienen un índice par.

Existen otras importantes pseudo clases relacionadas con esta última, como **first-child**, **last-child** y **only-child**, algunas de ellas recientemente incorporadas. La pseudo clase **first-child** referencia solo el primer hijo, **last-child** referencia solo el último hijo, y **only-child** afecta un elemento siempre y cuando sea el único hijo disponible. Estas pseudo clases en particular no requieren palabras clave o parámetros, y son implementadas como en el siguiente ejemplo:

```
*{
  margin: 0px;
}
p:last-child{
  background: #999999;
}
```

Listado 2-19. *Usando last-child para modificar solo el último elemento <p> de la lista.*

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

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

Listado 2-20. *Aplicando estilos a cada elemento, excepto <p>.*

La regla del Listado 2-20 asignará un margen de 0 pixeles a cada elemento del documento excepto los elementos **<p>**. A diferencia del selector universal utilizado previamente, la pseudo clase **not()** nos permite declarar una excepción. Los estilos en la regla creada con esta pseudo clase serán asignados a todo elemento excepto aquellos incluidos en la referencia entre paréntesis. En lugar de la palabra clave de un elemento podemos usar cualquier otra referencia que deseemos. En el próximo listado, por ejemplo, todos los elementos serán afectados excepto aquellos con el valor **mitexto2** en el atributo

class:

```
:not(.mitexto2){
  margin: 0px;
}
```

Listado 2-21. *Excepción utilizando el atributo class.*

Cuando aplicamos la última regla al código HTML del Listado 2-14 el navegador asigna los estilos por defecto al elemento **<p>** identificado con el atributo **class** y el valor **mitexto2** y provee un margen de 0 pixeles al resto.

Nuevos selectores

Hay algunos selectores más que fueron agregados o que ahora son considerados parte de CSS3 y pueden ser útiles para nuestros diseños. Estos selectores usan los símbolos **>**, **+** y **~** para especificar la relación entre elementos.

```
div > p.mitexto2{
  color: #990000;
}
```

Listado 2-22. *Selector >.*

El selector **>** está indicando que el elemento a ser afectado por la regla es el elemento de la derecha cuando tiene al de la izquierda como su padre. La regla en el Listado 2-22 modifica los elementos **<p>** que son hijos de un elemento ** **. En este caso, fuimos bien específicos y referenciamos solamente el elemento **<p>** con el valor **mitexto2** en su atributo **class**.

El próximo ejemplo construye un selector utilizando el símbolo **+**. Este selector referencia al elemento de la derecha cuando es inmediatamente precedido por el de la izquierda. Ambos elementos deben compartir el mismo padre:

```
p.mitexto2 + p{
  color: #990000;
}
```

Listado 2-23. *Selector +.*

La regla del Listado 2-23 afecta al elemento **<p>** que se encuentra ubicado luego de otro elemento **<p>** identificado con el valor **mitexto2** en su atributo **class**. Si abre en su navegador el archivo HTML con el código del Listado 2-14, el texto en el tercer elemento **<p>** aparecerá en la pantalla en color rojo debido a que este elemento **<p>** en particular está posicionado inmediatamente después del elemento **<p>** identificado con el valor **mitexto2** en su atributo **class**.

El último selector que estudiaremos es el construido con el símbolo **~**. Este selector es similar al anterior pero el elemento afectado no necesita estar precediendo de inmediato al elemento de la izquierda. Además, más de un elemento puede ser afectado:

```
p.mitexto2 ~ p{
  color: #990000;
}
```

Listado 2-24. *Selector ~.*

La regla del Listado 2-24 afecta al tercer y cuarto elemento **<p>** de nuestra plantilla de ejemplo. El estilo será aplicado a todos los elementos **<p>** que son hermanos y se encuentran luego del elemento **<p>** identificado con el valor **mitexto2** en su atributo **class**. No importa si otros elementos se encuentran intercalados, los elementos **<p>** en la tercera y cuarta posición aún serán afectados. Puede verificar esto último insertando un elemento **mitexto** luego del elemento **<p>** que tiene el valor **mitexto2** en su atributo **class**. A pesar de este cambio solo los elementos **<p>** serán modificados por esta regla.