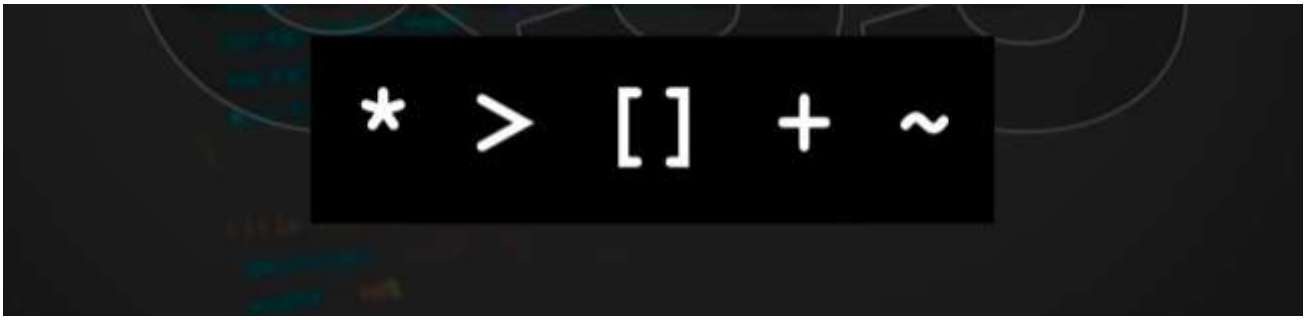
Docs_

Selectores



1 * (star)

```
* {
 margin: 0;
 padding: 0;
}
```

Comencemos con los más obvios, para los principiantes, antes de continuar con selectores más avanzados.

El asterisco se centrará en cada uno de los elementos en la página. Muchos desarrolladores usarán éste truco para poner a cero el `margin` y `padding`. . Mientras que esto está ciertamente bien para pruebas rápidas, yo te recomendaría que nunca uses esto en tu código final. Esto añade mucho *peso* en el navegador y es innecesario.

El `*` puede ser usado también con selectores de hijos.

```
#container * {
 border: 1px solid black;
}
```

Esto se centrará en cada uno de los elementos que sea hijo del `#container` `div`. . De nuevo, trata de no usar mucho ésta técnica, si no es que nunca.

Ejemplo

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
```

```
        * {
            border: 1px dotted black;
        }
    </style>
</head>
<body>
    <div>
        <p> My paragraph here. </p>
        <ul>
            <li> List Item 1</li>
            <li> List Item 2</li>
        </ul>
        <ul>
            <li> List Item 3</li>
            <li> List Item 4</li>
        </ul>
    </div>
</body>
```

Compatibilidad

- IE6+
- Firefox
- Chrome
- Safari
- Opera

2. #X (id)

```
#container {
 width: 960px;
 margin: auto;
}
```

Prefijar el símbolo numérico a un selector nos permite seleccionar por `id`. T. Éste es fácilmente el uso más común, sin embargo te cuidado cuando uses selectores `id` .

Pregúntate a ti mismo: ¿necesito absolutamente aplicar un `id` a este elemento para afectarlo?

Los selectores `id` son rígidos y no pueden ser re-utilizados. De ser posible, primero trata de usar un tag name, uno de los nuevos elementos en HTML5, incluso una pseudo-clase.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
      #container {
         background: #e3e3e3;
      }
    </style>
</head>
<body>
<div id="container">
   <p> My paragraph here. </p>
   <ul>
```

```
      <li> List Item 1</li>
      <li> List Item 2</li>
   </ul>
   <ul>
      <li> List Item 3</li>
      <li> List Item 4</li>
   </ul>
</div>
</body>
</html>
```

Compatibilidad

IE6+, Firefox,Chrome,Safari,Opera

3. .X (class)

```
.error {
color: red;
}
```

Éste es un selector de clase `class` La diferencia entre `id` s y `class` es que, con el último, puedes seleccionar varios elementos. Usa `class` es cuando quieras que un estilo aplique a un grupo de elementos. De manera alternativa, usa `id` s para encontrar una aguja en un pajar y estilizar sólo ése elemento en específico.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
      .warning {
         color: red;
      }
    </style>
</head>
<body>

<div id="container">
   <p> My paragraph here. </p>
   <ul>
      <li> List Item 1</li>
      <li> List Item 2</li>
   </ul>
```

```
   <ul>
      <li class="warning">Something went Wrong </li>
      <li> List Item 4</li>
   </ul>
</div>


</body>
</html>
```

Compatibilidad

- IE6+, Firefox,Chrome,Safari,Opera

4. X Y (descend)

```
li a {

text-decoration: none;

}
```

El siguiente selector más común es el selector `descendiente`. Cuando necesites ser más específico con tus selectores, usas éstos. Por ejemplo, ¿que tal sí, en lugar de seleccionar *todas* las etiquetas anchor, sólo necesitas los anchors que están dentro de una unordered list?. Aquí es cuando usarías específicamente un selector descendiente.

Pro-tip - Si tu selector luce como `X Y Z A B.error`, you're doing it wrong. lo estás haciendo mal.

Siempre pregúntate si es absolutamente necesario aplicar todo ese *peso*.

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8" />

    <title>CSS Selectors</title>

    <style>

       div p {

         color: red;

       }

    </style>

</head>

<body>


<div id="container">

   <p> My paragraph here. </p>

   <ul>

      <li> List Item 1</li>
```

```
      <li> List Item 2</li>
   </ul>


   <ul>
      <li class="warning">Something went Wrong </li>
      <li> List Item 4</li>
   </ul>
</div>


<p> Paragraph outside of div </p>
</body>
</html>
```

Compatibilidad IE6+ Firefox Chrome Safari Opera

5. X (tagName)

```
a { color: red; }

ul { margin-left: 0; }
```

¿Qué pasa si quieres seleccionar todos los elementos en una página, de acuerdo a su tipo (`type`), en vez de un nombre de `id` o `clase`. Mantenlo simple, y usa un selector de tipo. Si necesitas seleccionar todas las unordered lists, usa `ul {}`.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
       div {
         border: 1px solid black;
       }
    </style>
</head>
<body>


<div id="container">
   <p> My paragraph here. </p>
   <ul>
      <li> List Item 1</li>
      <li> List Item 2</li>
   </ul>


   <ul>
```

```
      <li class="warning">Something went Wrong </li>
      <li> List Item 4</li>
   </ul>
</div>

<p> Paragraph outside of div </p>
</body>
</html>
```

Compatibilidad IE6+ Firefox Chrome Safari Opera

6. X:visited and X:link (links)

```
a:link { color: red; }

a:visted { color: purple; }
```

Usamos la pseudo clase `:link` para seleccionar todas las etiquetas anchor a las que aún no se les ha dado click.

De manera alternativa, también tenemos la pseudo-clase `:visited`, la cual, tal como esperabas, nos permite aplicar un estilo específico sólo a las etiquetas anchor en la página *a las que se les ha dado* click, o han sido *visitadas*.

```
<!DOCTYPE html>
<html lang="en">


<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        a:link {
            color: red;
        }


        a:visted {
            color: purple;
        }
    </style>
</head>


<body>
```

```
    Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed do
eiusmod <a

        href="http://net.tutsplus.com">tempor</a> incididunt ut labore et dolore magna
aliqua. Ut enim ad minim veniam,

    quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

</body>



</html>
```

Compatibilidad IE7+ Firefox Chrome Safari Opera

7. X + Y (adjacent)

```
ul + p {
 color: red;
}
```

Este se conoce como un selector adyacente. Seleccionará *solamente* el elemento que es inmediatamente precedido por el primer elemento. En éste caso, solo el primer párrafo después de cada `ul` tendrá texto rojo.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        ul+p {
            color: red;
        }
    </style>
</head>

<body>

    <div id="container">
        <ul>
            <li> List Item </li>
            <li> List Item </li>
            <li> List Item </li>
```

```
            <li> List Item </li>

        </ul>



        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>

    </div>



</body>


</html>
```

Compatibilidad IE7+ Firefox Chrome Safari Opera

8. X > Y (childcombinator)

```
div#container > ul {

border: 1px solid black;

}
```

La diferencia entre el estándar `X Y` y `X > Y` es que el último sólo seleccionará hijos directos. Por ejemplo, considera el siguiente código.

```
<div id="container">
  <ul>
    <li> List Item
      <ul>
        <li> Child </li>
      </ul>
    </li>
    <li> List Item </li>
    <li> List Item </li>
    <li> List Item </li>
  </ul>
</div>
```

Un selector de `#container > ul` solo afectará los `ul`s que sean hijos directos del `div` con `id` de `container`. No afectará, por ejemplo, el `ul` que sea hijo del primer `li`.

Por ésta razón, hay beneficios de desempeño por usar el elemento de hijo. De hecho, es recomendable particularmente cuando se trabaja con motores de selectores CSS basados en JavaScript.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        #container>ul {
            border: 1px solid black;
        }
    </style>
</head>

<body>

    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
            <li> List Item </li>
            <li> List Item </li>
        </ul>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>
```

```
        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>

    </div>



</body>

</html>
```

Compatibilidad

- IE7+
- Firefox
- Chrome
- Safari
- Opera

9. X ~ Y (generalcombinator)

```
ul ~ p {
 color: red;
}
```

Este elemento “hermano” es similar a `X + Y`, sin embargo, es menos estricto. Mientras que un selector adyacente (`ul + p`) sólo selecciona el primer elemento que es *inmediatamente* precedido por el primer selector, éste es más generalizado. Seleccionará, refiriéndonos al ejemplo de arriba, cualquier elemento `p`, siempre y cuando estén después de un `ul`.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        ul~p {
            color: red;
        }
    </style>
</head>
<body>
    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
```

```
            <li> List Item </li>

            <li> List Item </li>

            <li> List Item </li>

        </ul>

        <p> Lorem ipsum dolor sit amet, <a href="#" title="Some title">consectetur</a>
adipisicing elit, sed do eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit</p>

    </div>

</body>

</html>
```

Compatibilidad IE7+ Firefox Chrome Safari Opera

10. X[title] (attributes)

```
a[title] {

 color: green;

}
```

Denominado como un *selector de atributos*, en nuestro ejemplo de arriba, esto sólo seleccionará las etiquetas anchor que tengan un atributo `title`. Las etiquetas anchor que no lo tengan no recibirán éste estilo en particular. Pero,¿ y qué si necesitas ser más específico? Bueno…

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        a[title] {
            color: green;
        }
    </style>
</head>
<body>
    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
```

```
            <li> List Item </li>

            <li> List Item </li>

        </ul>

        <p> Lorem ipsum dolor sit amet, <a href="#" title="Some title">consectetur</a>
adipisicing elit, sed do eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do. </p>

    </div>

</body>

</html>
```

Compatibilidad IE7+ Firefox Chrome Safari Opera

11. X[href="foo"] (attributes2)

```
a[href="https://net.tutsplus.com"] {

color: #1f6053; /* nettuts green */

}
```

El fragmento de arriba dará estilo a todas las etiquetas anchor que enlacen a *http://net.tutsplus.com*; estas recibirán nuestro característico color verde. Las demás etiquetas anchor permanecerán sin cambio.

Ten en cuenta que estamos encerrando el valor entre comillas. Recuerda hacer esto también cuando uses un motor de selectores CSS JavaScript. Cuando sea posible, siempre usa selectores CSS3 en vez de métodos no oficiales.

Ésto funciona bien, aunque, es un poco rígido. ¿Que tal si el enlace dirige ciertamente a Nettuts+, pero, tal vez, la dirección es *nettuts.com* en lugar de la url completa? En esos casos, podemos usar un poco de sintaxis de expresiones regulares.

```
<!DOCTYPE html>

<html lang="en">


<head>

    <meta charset="UTF-8" />

    <title>CSS Selectors</title>

    <style>

        a[href="http://net.tutsplus.com"] {

            color: #1f6053;

        }

    </style>

</head>


<body>
```

```
    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
            <li> List Item </li>
            <li> List Item </li>
        </ul>

        <p> Lorem ipsum dolor sit amet, <a href="#" title="Some title">consectetur</a>
adipisicing elit, sed do <a
                href="http://nettuts.com">Nettuts</a> tempor. </p>
        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>
        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>
        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur adipisicing elit, sed do
            eiusmod tempor. </p>
    </div>

</body>

</html>
```

Compatibilidad IE7+ Firefox Chrome Safari Opera

12. X[href*="nettuts"] (attributes3)

```
a[href*="tuts"] {

color: #1f6053; /* nettuts green */

}
```

Aquí vamos; eso es lo que necesitamos. El asterisco designa que el valor a continuación debe aparecer *en algún lugar* del valor del atributo. De esa manera, esto cubre *nettuts.com*, *net.tutsplus.com,* e incluso *tutsplus.com*.

Ten en mente que esta es una declaración abierta. ¿Qué si la etiqueta anchor enlazara a algún sitio no perteneciente a Envato con la palabra *tuts* en la url? Cuando necesitas ser más específico, usa `^` y `$`, para hacer referencia al principio y fin de una cadena de texto, respectivamente.

```
<!DOCTYPE html>
<html lang="en">


<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        a[href*="tuts"] {
            color: #1f6053;
            /* nettuts green */
        }
    </style>
</head>


<body>


    <div id="container">
```

```
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
            <li> List Item </li>
            <li> List Item </li>
        </ul>

        <p> Lorem ipsum dolor sit amet, <a href="#" title="Some title">consectetur</a>
adipisicing elit, sed do <a
                href="http://nettuts.com">Nettuts</a> tempor. </p>
        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>
        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>
        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur adipisicing elit, sed do
            eiusmod tempor. </p>
    </div>

</body>

</html>
```

Compatibilidad

- IE7+ Firefox Chrome Safari Opera

13. X[href^="http"] (attributes4)

```
a[href^="http"] {
 background: url(path/to/external/icon.png) no-repeat;
 padding-left: 10px;
}
```

¿Alguna vez te has preguntado cómo es que algunos sitios son capaces de desplegar un pequeño icono junto a los enlaces que son externos? Estoy seguro de que los has visto antes; son buenos recordatorios de que el enlace te dirigirá a un sitio web totalmente diferente.

Esto es un juego de niños con el símbolo de intercalación. Es más comúnmente usado en expresiones regulares para designar el inicio de una cadena de texto. Si queremos afectar todas las etiquetas anchor que tienen un `href` que comienza con `http`, podríamos usar un selector similar al fragmento mostrado arriba.

Observa que no estamos buscando `http://`; eso es innecesario y no cuenta para las urls que comienzan con `https://`.

Ahora, ¿si en vez de eso quisiéramos aplicar estilo a todas las etiquetas anchor que enlacen a, por decir, una foto? En esos casos, busquemos el *final* de la cadena de texto.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        a[href^="http"] {
            color: red;
        }
    </style>
```

```
</head>

<body>

    <div id="container">

        <ul>

            <li> List Item

                <ul>

                    <li> Child </li>

                </ul>

            </li>

            <li> List Item </li>

            <li> List Item </li>

            <li> List Item </li>

        </ul>

        <p> Lorem ipsum dolor sit amet, <a href="#" title="Some title">consectetur</a>
adipisicing elit, sed do <a

                href="http://nettuts.com">Nettuts</a> tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>

        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur adipisicing elit, sed do

            eiusmod tempor. </p>

    </div>

</body>

</html>
```

Compatibilidad

- IE7+ Firefox Chrome Safari Opera

14. X[href$=".jpg"] (attributes5)

```
a[href$=".jpg"] {
 color: red;
}
```

De nuevo, usamos un símbolo de expresiones regulares, `$`, para referirnos al final de una cadena de texto. En éste caso, estamos buscando todos los anchor que enlacen a una imagen – o por lo menos a una url que termine con `.jpg`. Ten en mente que esto seguramente no funcionará para `gifs` ni `pngs`.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        a[href$=".jpg"] {
            color: red;
        }
    </style>
</head>

<body>

    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
```

```
                </ul>

            </li>

            <li> List Item </li>

            <li> List Item </li>

            <li> List Item </li>

        </ul>


        <p> Lorem ipsum dolor sit amet, <a href="#" title="Some title">consectetur</a>
adipisicing elit, sed do <a

                href="http://nettuts.com">Nettuts</a> tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>

        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur "<a

                href="http://d2o0t5hpnwv4c1.cloudfront.net/839_git/preview.jpg">Getting
Good with Git</a>" elit, sed do

            eiusmod tempor. </p>

    </div>


</body>


</html>
```

Compatibilidad

- IE7+ Firefox Chrome Safari Opera

15. X[data-*="foo"] (attributes6)

```
a[data-filetype="image"] {

 color: red;

}
```

Regresemos al número ocho; ¿como compensamos para los diferentes tipos de imagen: `png`, `jpeg,jpg`, `gif`? Bueno, podríamos crear múltiples selectores, tal como:

```
a[href$=".jpg"],

a[href$=".jpeg"],

a[href$=".png"],

a[href$=".gif"] {

 color: red;

}
```

Pero, eso es un dolor en el trasero y es ineficiente. Otra posible solución es usar atributos personalizados. ¿Y si agregáramos nuestro propio atributo `data-filetype` para cada anchor que enlaza a una imagen?

```
<a href="path/to/image.jpg" data-filetype="image"> Image Link </a>
```

Entonces, *dicho esto*, dicho esto, podemos usar un selector de atributos estándar para afectar sólo a esos anchors.

```
a[data-filetype="image"] {

 color: red;

}
```

Compatibilidad

- IE7+ Firefox Chrome Safari Opera

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        a[data-filetype="image"] {
            color: red;
        }
    </style>
</head>

<body>

    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
            <li> List Item </li>
            <li> List Item </li>
        </ul>
```

```
        <p> Lorem ipsum dolor sit amet, <a href="#" title="Some title">consectetur</a>
adipisicing elit, sed do <a

                href="http://nettuts.com">Nettuts</a> tempor. </p>

        <p> <a href="http://d2o0t5hpnwv4c1.cloudfront.net/817_rubyNewbies1/preivew.png"
data-filetype="image">PNG

                Image</a> ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>

        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur "<a

                href="http://d2o0t5hpnwv4c1.cloudfront.net/839_git/preview.jpg" data-
filetype="image">Getting Good with

                Git</a>" elit, sed do eiusmod tempor. </p>

    </div>



</body>



</html>
```

16. X[foo~="bar"] (attributes7)

```
a[data-info~="external"] {
 color: red;
}
a[data-info~="image"] {
 border: 1px solid black;
}
```

Aquí hay uno especial que impresionará a tus amigos. No mucha gente sabe de éste truco. El símbolo tilde (~) nos permite afectar un atributo que tenga una lista de valores separados por espacios.

Siguiendo con nuestro atributo personalizado del número 15, arriba, podríamos crear un atributo `data-info` el cual puede recibir una lista separada por espacios de lo que sea que necesitemos tomar nota. En éste caso, haremos nota de enlaces externos y enlaces a imágenes – justo para el ejemplo.

```
<a href="path/to/image.jpg" data-info="external image"> Click Me, Fool </a>
```

Con ése código en su lugar, ahora podemos afectar cualquier etiqueta que tenga cualquiera de esos valores, usando el truco de selector de atributos con ~.

```
/* Target data-info attr that contains the value "external" */
a[data-info~="external"] {
 color: red;
}
/* And which contain the value "image" */
a[data-info~="image"] {
border: 1px solid black;
}
```

Compatibilidad IE7+ Firefox Chrome SafariOpera

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        a[data-info~="external"] {
            color: red;
        }

        a[data-info~="image"] {
            border: 1px solid black;
        }
    </style>
</head>

<body>

    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
            <li> List Item </li>
```

```
            <li> List Item </li>

        </ul>



        <p> Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. </p>

        <p> <a href="http://d2o0t5hpnwv4c1.cloudfront.net/817_rubyNewbies1/preivew.png"
data-filetype="image">PNG

                Image</a> ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>

        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur "<a

                href="http://d2o0t5hpnwv4c1.cloudfront.net/839_git/preview.jpg" data-
info="external image">Getting Good

                with Git</a>" elit, sed do eiusmod tempor. </p>

    </div>


</body>


</html>
```

17. X:checked (checked)

```
input[type=radio]:checked {
 border: 1px solid black;
}
```

Ésta pseudo clase sólo afectará a un elemento de interfaz de usuario que haya sido *seleccionado* - como un botón de opción (radio button) o casilla de selección (checkbox). Tan sencillo como eso.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
       input[type=radio]:checked + label {
          color: blue;
       }
    </style>
</head>
<body>
   <form>
      <input type="radio" name="rad" value="Radio Button">
      <label for="rad"> Radio Button</label>
   </form>
</body>
</html>
```

Compatibilidad IE9+ Firefox Chrome Safari Opera

18. X:after

Las pseudo clases `before` y `after` son lo máximo. Todos los días, al parecer, la gente está encontrando nuevas y creativas formas para usarlas de manera efectiva. Éstas simplemente generan contenido alrededor del elemento seleccionado. Muchos conocieron primero éstas clases cuando encontraron el hack de clear-fix.

```
.clearfix:after {
 content: "";
 display: block;
 clear: both;
 visibility: hidden;
 font-size: 0;
 height: 0;
 }
.clearfix {
 *display: inline-block;
 _height: 1%;
}
```

Éste *hack* usa la pseudo clase `:after` para agregar un espacio después del elemento, y luego borrarlo. Es un excelente truco para tener en tu bolsa de herramientas, particularmente en los casos cuando el método `overflow: hidden;` no es posible.

De acuerdo con las especificaciones de Selectores CSS3, técnicamente deberías usar la sintaxis del pseudo elemento de dos `::`. Sin embargo, para permanecer compatible, el agente-usuario aceptará un solo : también. De hecho, en éste punto, es más inteligente un solo : en tus proyectos.

- IE8+ Firefox Chrome Safari Opera

19. X:hover

```
div:hover {
background: #e3e3e3;
}
```

El término oficial para este es `pseudo clase de acción de usuario`. Suena confuso, pero en realidad no lo es. ¿Quieres aplicar un estilo específico cuando un usuario pasa sobre un elemento? ¡Esto hará el trabajo!

Ten en cuenta que versiones más viejas de Inernet Explorer no responden cuando la pseudo clase `:hover` es aplicada a cualquier otra cosa que no sea una etiqueta anchor.

Usarás más seguido éste selector cuando apliques, por ejemplo, un `border-bottom` a etiquetas anchor, cuando se pase por encima.

```
a:hover {
 border-bottom: 1px solid black;
}
```

Pro-tip - `border-bottom: 1px solid black;` luce mejor que `text-decoration: underline;`.

Compatibilidad

- IE6+ (In IE6, :hover must be applied to an anchor element)
- Firefox
- Chrome
- Safari
- Opera

20. X:not(selector) (not)

```
div:not(#container) {
 color: blue;
}
```

La pseudo clase `negación` es particularmente útil. Digamos que quiero seleccionar todos los divs, excepto por los que tienen un `id` de `container`. El fragmento de arriba manejará la tarea perfectamente.

O, si hubiera querido seleccionar todos los elementos (no aconsejable) excepto por las etiquetas de párrafo, podríamos hacer:

```
*:not(p) {
color: green;
}
```

Compatibilidad

- IE9+
- Firefox
- Chrome
- Safari
- Opera

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        div:not(#container) {
            color: blue;
```

```
        }

        *:not(p) {
            color: green;
        }
    </style>
</head>

<body>

    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
            <li> List Item </li>
            <li> List Item </li>
        </ul>

        <ul>
            <li>
                <a href="#"> Anchor Tag </a>
            </li>
            <li>
                <a href="#"> Anchor Tag </a>
```

```
            </li>

            <li>

                <a href="#"> Anchor Tag </a>

            </li>

            <li>

                <a href="#"> Anchor Tag </a>

            </li>

        </ul>


        <p> Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-info="external" title="Some title">consectetur</a> adipisicing elit, sed do <a href="http://nettuts.com">Nettuts</a> tempor. </p>

        <p> <a href="http://d2o0t5hpnwv4c1.cloudfront.net/817_rubyNewbies1/preivew.png" data-filetype="image">PNG Image</a> ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed do eiusmod tempor. </p>

        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet, consectetur "<a href="http://d2o0t5hpnwv4c1.cloudfront.net/839_git/preview.jpg" data-info="external image">Getting Good with Git</a>" elit, sed do eiusmod tempor. </p>

    </div>

    <div> Lorem 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. </div>


</body>


</html>
```

21. X::pseudoElement (pseudoElements)

```
p::first-line {
 font-weight: bold;
 font-size: 1.2em;
}
```

Podemos usar pseudo elementos (designados por `::`) para estilizar fragmentos de un elemento, como la primera línea, o la primera letra. Ten en mente que estos deben ser aplicados a elementos de nivel bloque para que funcione.

Un pseudo-elemento está compuesto de dos “dos puntos” `::`

Afecta La Primer Letra De Un Párrafo

```
p::first-letter {
 float: left;
 font-size: 2em;
 font-weight: bold;
 font-family: cursive;
 padding-right: 2px;
}
```

Este fragmento de código es una abstracción que encontrará todos los párrafos en la página, y luego afectará sólo la primer letra de ése elemento.

Esto es más usado para crear estilos como el de los periódicos para la primer letra de un artículo.

Afecta La Primer Linea De Un Párrafo

```
p::first-line {
 font-weight: bold;
```

```
font-size: 1.2em;

}
```

Similarmente, el pseudo elemento `::first-line` estilizará, como es esperado, sólo la primer línea del elemento.

> “Para compatibilidad con hojas de estilo existentes, los agentes de usuario deben aceptar también la notación previa de un “dos puntos” para pseudo-elementos introducidos en niveles 1 y 2 de CSS (es decir, :first-line, :first-letter, :before and :after).Esta compatibilidad no está permitida para los nuevos pseudo-elementos introducidos en ésta especificación.” - Fuente

Compatibilidad

- IE6+
- Firefox
- Chrome
- Safari
- Opera

```
<!DOCTYPE html>
<html lang="en">


<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        p::first-line {
            font-weight: bold;
            font-size: 1.2em;
        }

        p::first-letter {
```

```
            float: left;
            font-weight: bold;
            font-family: cursive;
            font-size: 2em;
            padding-right: 2px;
        }
    </style>
</head>

<body>

    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
            <li> List Item </li>
            <li> List Item </li>
        </ul>

        <ul>
            <li>
                <a href="#"> Anchor Tag </a>
            </li>
            <li>
```

```
                <a href="#"> Anchor Tag </a>

            </li>

            <li>

                <a href="#"> Anchor Tag </a>

            </li>

            <li>

                <a href="#"> Anchor Tag </a>

            </li>

        </ul>



        <p> Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. </p>

        <p> <a href="http://d2o0t5hpnwv4c1.cloudfront.net/817_rubyNewbies1/preivew.png"
data-filetype="image">PNG

                Image</a> ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>
```

```
        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur "<a

                href="http://d2o0t5hpnwv4c1.cloudfront.net/839_git/preview.jpg" data-
info="external image">Getting Good

                with Git</a>" elit, sed do eiusmod tempor. </p>

    </div>



    <div> Lorem 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. </div>



</body>



</html>
```

22. X:nth-child(n) (nth)

```
li:nth-child(3) {
 color: red;
}
```

¿Recuerdas los días cuando no teníamos manera de afectar elementos específicos en un montón? ¡La pseudo clase `nth-child` resuelve eso!

Por favor observa que `nth-child` acepta un número entero como parámetro, sin embargo, éste no es basado en cero. Si quieres afectar el segundo elemento de una lista, usa `li:nth-child(2)`.

Podemos incluso usar esto para seleccionar un conjunto variable de hijos. Por ejemplo, podríamos usar `li:nth-child(4n)` para seleccionar todos los cuartos elementos de una lista.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        li:nth-child(2) {
            color: red;
        }
    </style>
</head>

<body>

    <div id="container">
```

```
<ul>
    <li> List Item
        <ul>
            <li> Child </li>
            <li> Child </li>
            <li> Child </li>
        </ul>
    </li>
    <li> List Item </li>
    <li> List Item </li>
    <li> List Item </li>
</ul>

<ul>
    <li>
        <a href="#"> Anchor Tag </a>
    </li>
    <li>
        <a href="#"> Anchor Tag </a>
    </li>
    <li>
        <a href="#"> Anchor Tag </a>
    </li>
    <li>
        <a href="#"> Anchor Tag </a>
    </li>
</ul>
```

```
        <p> Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. </p>

        <p> <a href="http://d2o0t5hpnwv4c1.cloudfront.net/817_rubyNewbies1/preivew.png"
data-filetype="image">PNG

                Image</a> ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>

        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur "<a

                href="http://d2o0t5hpnwv4c1.cloudfront.net/839_git/preview.jpg" data-
info="external image">Getting Good

                with Git</a>" elit, sed do eiusmod tempor. </p>

    </div>



    <div> Lorem 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. </div>


</body>


</html>
```

Compatibilidad

- IE9+
- Firefox 3.5+
- Chrome
- Safari

23. X:nth-last-child(n) (nthChild)

```
li:nth-last-child(2) {
 color: red;
}
```

¿Y si tuvieras una inmensa lista de elementos en un `ul`, y solo necesitaras accesar, digamos, del tercer al último elemento? En vez de usar `li:nth-child(397)`, en su lugar podrías usar la pseudo clase `nth-last-child`.

Ésta técnica funciona casi igual que el número dieciséis, sin embargo, la diferencia es que comienza al final del grupo y de ahí hacia atrás.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        li:nth-last-child(2) {
            color: red;
        }
    </style>
</head>
<body>
    <div id="container">
        <ul>
            <li> List Item
                <ul>
                    <li> Child </li>
```

```
                <li> Child </li>
                <li> Child </li>
            </ul>
        </li>
        <li> List Item </li>
        <li> List Item </li>
        <li> List Item </li>
    </ul>
    <ul>
        <li>
            <a href="#"> Anchor Tag </a>
        </li>
        <li>
            <a href="#"> Anchor Tag </a>
        </li>
        <li>
            <a href="#"> Anchor Tag </a>
        </li>
        <li>
            <a href="#"> Anchor Tag </a>
        </li>
    </ul>
    <p> Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"
            title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>
        tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"
            title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>
```

```
            tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. Lorem ipsum dolor sit amet, <a href="http://www.google.com" data-
info="external"

                title="Some title">consectetur</a> adipisicing elit, sed do <a
href="http://nettuts.com">Nettuts</a>

            tempor. </p>

        <p> <a href="http://d2o0t5hpnwv4c1.cloudfront.net/817_rubyNewbies1/preivew.png"
data-filetype="image">PNG

                Image</a> ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor. </p>

        <p> Lorem ipsum dolor sit amet, consectetur <a href="#">adipisicing</a> elit, sed
do eiusmod tempor. </p>

        <p> Lorem ipsum <a href="http://net.tutsplus.com">Nettuts+</a> sit amet,
consectetur "<a

                href="http://d2o0t5hpnwv4c1.cloudfront.net/839_git/preview.jpg" data-
info="external image">Getting Good

                with Git</a>" elit, sed do eiusmod tempor. </p>

    </div>



    <div> Lorem 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. </div>



</body>



</html>
```

24. X:nth-of-type(n) (nthOfType)

```
ul:nth-of-type(3) {
 border: 1px solid black;
}
```

Habrá veces cuando, en vez de seleccionar un `hijo`, necesites seleccionar de acuerdo al tipo (`type`) de elemento.

Imagina un código que contiene cinco listas sin orden. Si quisieras estilizar sólo la tercera `ul`, y no tuvieras un `id` único al cual engancharla, podrías usar la pseudo clase `nth-of-type(n)` En el fragmento de arriba, sólo el tercer `ul` tendrá un borde a su alrededor.

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
        ul:nth-of-type(3) {
            border: 1px solid black;
        }
    </style>
</head>

<body>

    <div id="container">
        <ul>
```

```
            <li> List Item
                <ul>
                    <li> Child </li>
                    <li> Child </li>
                    <li> Child </li>
                </ul>
            </li>
            <li> List Item </li>
            <li> List Item </li>
            <li> List Item </li>
        </ul>

        <ul>
            <li>
                <a href="#"> Anchor Tag </a>
            </li>
            <li>
                <a href="#"> Anchor Tag </a>
            </li>
            <li>
                <a href="#"> Anchor Tag </a>
            </li>
            <li>
                <a href="#"> Anchor Tag </a>
            </li>
        </ul>

        <ul>
```

```
            <li> List Items </li>

            <li> List Items </li>

            <li> List Items </li>

        </ul>

</body>


</html>
```

Compatibilidad

- IE9+
- Firefox 3.5+
- Chrome
- Safari

25. X:nth-last-of-type(n)

```
ul:nth-last-of-type(3) {

 border: 1px solid black;

}
```

Y sí, para permanecer consistente, también podemos usar `nth-last-of-type` para comenzar al final de la lista del selector y partir hacia atrás para afectar al elemento deseado.

Compatibilidad

- IE9+
- Firefox 3.5+
- Chrome
- Safari
- Opera

26. X:first-child (firstChild)

```
ul li:first-child {
 border-top: none;
}
```

Esta pseudo clase estructural nos permite afectar sólo al primer hijo del padre del elemento. Usarás esto frecuentemente para remover bordes de los primeros y últimos elementos de una lista.

Por ejemplo, digamos que tienes una lista de filas, y cada una tiene un borde superior `border-top` y un borde inferior `border-bottom`. Bueno, con ése arreglo, el primer y último elemento en ese conjunto lucirán un poco raro.

Muchos diseñadores aplican clases de `first` y `last` para compensar esto. En lugar de eso, puedes usar éstas pseudo clases.

Compatibilidad

- IE7+
- Firefox
- Chrome
- Safari
- Opera

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>

       ul {
         width: 200px;
```

```
        background: #292929;
        color: white;
        list-style: none;
        padding-left: 0;
      }
      li {
        padding: 10px;
        border-bottom: 1px solid black;
        border-top: 1px solid #3c3c3c;
      }
 li:first-child {
    border-top: none;
}
li:last-child {
   border-bottom: none;
}
    </style>
</head>
<body>
   <div id="container">
      <ul>
        <li> List Item </li>
        <li> List Item </li>
        <li> List Item </li>
      </ul>
</body>
</html>
```

27. X:last-child

```
ul > li:last-child {
 color: green;
}
```

Al contrario de `first-child`, `last-child` afectará el último elemento del padre el elemento.

Ejemplo

Contruyamos un ejemplo simple para demostrar uno posible uso de estas clases. Crearemos un elemento de lista estilizado.

Código

```
<ul>
  <li> List Item </li>
  <li> List Item </li>
  <li> List Item </li>
</ul>
```

Nada especial aquí, sólo una simple lista.

CSS

```
ul {
 width: 200px;
 background: #292929;
 color: white;
 list-style: none;
 padding-left: 0;
```

```
}


li {
 padding: 10px;
 border-bottom: 1px solid black;
 border-top: 1px solid #3c3c3c;
}
```

Éste estilo fijará un fondo, removerá el padding por defecto del navegador en la `ul`, y aplicará bordes para cada `li` para proveer un poco de profundidad.

Para agregar profundidad a tus listas, aplica un borde inferior `border-bottom` a cada `li` que es un tono o dos más oscuros que el color de fondo del `li`. Después, aplica un borde superior `border-top` que sea un par de tonos más *claro*.

El único problema, como se muestra en la imagen superior, es que el borde será aplicado la parte superior e inferior de la lista sin orden – lo cual luce un poco extraño. Usemos las pseudo clases `:first-child` y `:last-child` para arreglar esto.

```
li:first-child {
  border-top: none;
}
```

```
li:last-child {
 border-bottom: none;
}
```

Compatibilidad IE9+ Firefox Chrome Safari Opera

Sip – IE8 soporta `:first-child`*, pero* `:last-child`*. Imagínate.*

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>


       ul {
         width: 200px;
         background: #292929;
         color: white;
         list-style: none;
```

```
        padding-left: 0;
      }
      li {
        padding: 10px;
        border-bottom: 1px solid black;
        border-top: 1px solid #3c3c3c;
      }
 li:first-child {
    border-top: none;
}
li:last-child {
   border-bottom: none;
}
    </style>
</head>
<body>
   <div id="container">
      <ul>
        <li> List Item </li>
        <li> List Item </li>
        <li> List Item </li>
      </ul>
</body>
</html>
```

28. X:only-child (onlyChild)

```
div p:only-child {
 color: red;
}
```

A decir verdad, probablemente no te veas usando la pseudo clase `only-child` muy seguido. No obstante, está disponible, en caso de que la necesites.

Te permite afectar elementos que sean los *únicos* hijos de su padre. Por ejemplo, haciendo referencia al fragmento de código de arriba, sólo el párrafo que es el único hijo del `div` será coloreado rojo.

Asumamos el siguiente código.

```
<div><p> Mi Párrafo. </p></div>


<div>
 <p> Dos Párrafos en Total. </p>
 <p> Dos Párrafos en Total. </p>
</div>
```

En este caso, el párrafo del segundo `div` no será afectado; sólo el primer `div`. Tan pronto como apliques más de un hijo a un elemento, la pseudo clase `only-child` deja de tener efecto.

Compatibilidad

- IE9+
- Firefox
- Chrome
- Safari
- Opera

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
       div p:only-child {
         color: red;
       }
    </style>
</head>
<body>

<div><p> My paragraph here. </p></div>

<div>
   <p> Two paragraphs total. </p>
   <p> Two paragraphs total. </p>
</div>

</body>
</html>
```

29. X:only-of-type

```
li:only-of-type {
 font-weight: bold;
}
```

Esta pseudo clase estructural puede ser usada en maneras inteligentes. Afectará elementos que no tienen hermanos dentro de su contenedor padre. Como ejemplo, afectemos a todos los `ul`s, que tengan sólo un elemento de lista.

Primero, pregúntate cómo lograrías esta tarea. Podrías usar `ul li`, pero, esto afectaría a *todos* los elementos de lista. La única solución es usar `only-of-type`.

```
ul > li:only-of-type {
 font-weight: bold;
}
```

Compatibilidad IE9+ Firefox 3.5+ Chrome Safari Opera

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
       div p:only-of-type {
         color: red;
      }

      li:only-of-type {
         font-weight: bold;
```

```
      }
    </style>
</head>
<body>

<div>
   <p> My paragraph here. </p>
   <ul>
      <li> List Item </li>
      <li> List Item </li>
   </ul>
</div>

<div>
   <p> Two paragraphs total. </p>
   <p> Two paragraphs total. </p>
   <ul>
      <li> List Item </li>
   </ul>
</div>

</body>
</html>
```

30. X:first-of-type

La pseudo clase `first-of-type` te permite seleccionar los primeros hermanos de su tipo.

Una Prueba

Para entender mejor esto, hagamos una prueba. Copia el siguiente código en tu editor de texto:

```
<div>
 <p> My paragraph here. </p>
 <ul>
   <li> List Item 1 </li>
   <li> List Item 2 </li>
 </ul>

 <ul>
   <li> List Item 3 </li>
   <li> List Item 4 </li>
 </ul>
</div>
```

Ahora, sin leer más adelante, trata de resolver como afectar sólo a *"List Item 2"*. Cuando lo soluciones (o te rindas), continúa leyendo.

Solución 1

Hay una variedad de formas de resolver ésta prueba. Revisaremos un puñado de ellas. Comencemos usando `first-of-type`.

```
ul:first-of-type > li:nth-child(2) {
 font-weight: bold;
```

```
}
```

El código en esencia dice, “encuentra la primer lista sin orden en la página, después encuentra sólo el hijo inmediato, que son elementos de lista. Después, fíltralo a sólo el segundo elemento de la lista en ese conjunto.

Solución 2

Otra opción es usar el selector adyacente.

```
p + ul li:last-child {
 font-weight: bold;
}
```

En éste escenario, encontramos el `ul` que inmediatamente procede a la etiqueta `p` y luego encontramos el último hijo del elemento.

Solución 3

Podemos ser tan odiosos o traviesos como queramos con éstos selectores.

```
ul:first-of-type li:nth-last-child(1) {
 font-weight: bold;
}
```

Ésta vez, tomamos el primer `ul` en la página y despúes encontramos el primer elemento, ¡pero empezando desde el fondo! :)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>CSS Selectors</title>
    <style>
       ul:first-of-type > li:nth-child(2) {
```

```
        font-weight: bold;
      }
    </style>
</head>
<body>
<div>
   <p> My paragraph here. </p>
   <ul>
      <li> List Item 1</li>
      <li> List Item 2</li>
   </ul>
   <ul>
      <li> List Item 3</li>
      <li> List Item 4</li>
   </ul>
</div>
</body>
</html>
```