

Create professional videos instantly!

Create Now!



tuts+



Advertisement

CODE > WEB DEVELOPMENT

Os 30 Seletores CSS Que Você Deve Memorizar

by [Jeffrey Way](#) 9 Jun 2011Difficulty: Intermediate Length: Long Languages:

Web Development

Front-End

HTML

CSS

CSS Selectors



This post is part of a series called [CSS3 Mastery](#).

◀ [10 CSS3 Properties you Need to be Familiar With](#)

▶ [Getting to Work with CSS3 Power Tools](#)

Portuguese (Português) translation by [Erick Patrick](#) (you can also [view the original English article](#))

Então, você aprendeu os seletores básicos: `identificador`, `classe`, e `descendente` - e achou que tinha aprendido tudo? Se acha que sim, então você está deixando de aproveitar uma gama de possibilidades. Enquanto a maioria dos seletores mencionados nesse artigo fazem parte da especificação do CSS3, e estão, consequentemente, disponíveis apenas em navegadores modernos, você deve saber todos eles de cabeça.

1. *

```
1  * {  
2    margin: 0;  
3    padding: 0;  
4  }
```

Vamos falar primeiro dos mais óbvios, para os iniciantes, antes de começarmos a falar dos seletores mais avançados.

O asterísco (ou estrela) tem como alvo todos os elementos em uma página. Muitos desenvolvedores usam-no para zerar as margens e o preenchimento (`margin` e `padding`, respectivamente). Embora isso funcione para testes rápidos, recomendo não usar essa técnica em produção. Ela deixa o navegador muito *pesado* (por ter de selecionar todo e cada elemento), além de ser desnecessário.

O `*` também pode ser usado como um seletor filho.

```
1  #container * {  
2    border: 1px solid black;  
3  }
```

Isso mirará todo elemento que for filho de uma `div` com um `identificador` nomeado de `#container`. Novamente, não use muito essa técnica em produção. De preferência, nunca.

[Visualizar Demonstração](#)

Compatibilidade

- IE6+
- Firefox
- Chrome

- Safari
- Opera

Advertisement

2. #X

```
1 #container {  
2   width: 960px;  
3   margin: auto;  
4 }
```

Prefixar um seletor com o "jogo da velha" (ou cerquilha, caso prefira), permite mirar algum elemento com um `identificador`. Ele é um dos seletores mais usados, contudo, seja cuidadoso ao usar seletores com `identificador`.

Faça-se a seguinte pergunta: Preciso, realmente, aplicar um `identificador` a esse elemento para poder selecioná-lo

Seletores `identificador` são rígidos e não permitem reuso. Se possível, tente usar, primeiro, o nome da tag, um dos novos elementos do HTML5 ou uma pseudo-classe.

[Visualizar Demonstração](#)

Compatibilidade

- IE6+
- Firefox
- Chrome

- Safari
- Opera

3. .X

```
1 | .error {  
2 |   color: red;  
3 | }
```

Esse é um seletor de `classe`. A diferença entre um `identificador` e uma `classe` é que, com a classe, você pode selecionar vários elementos. Use uma `classe` quando você precisar estilizar um grupo de elementos. Alternativamente, use um `identificador` quando precisar achar "uma agulha num palheiro", para estilizar um único elemento.

[Visualizar Demonstração](#)

Compatibilidade

- IE6+
- Firefox
- Chrome
- Safari
- Opera

4. X Y

```
1 | li a {  
2 |   text-decoration: none;  
3 | }
```

O próximo seletor mais comum é o seletor de `descendência`. Quando precisar ser mais específico em seus seletores, use esse. Por exemplo, se, ao invés de mirar todas as tags de âncora (`a`), talvez você só precise selecionar aquelas que estão dentro de uma lista não ordenada? Esse seletor, o seletor descendente, é especificamente para isso.

Dica - Se seu seletor parecer com isso `X Y Z A B.error`, tenha certeza que há algo errado. Pergunte-se se,

realmente, é necessário todo esse peso, toda essa especificidade.

[Visualizar Demonstração](#)

Compatibilidade

- IE6+
- Firefox
- Chrome
- Safari
- Opera

5. X

```
1 | a { color: red; }  
2 | ul { margin-left: 0; }
```

E se você quiser selecionar todos os elementos da página, de acordo com o `tipo` deles, ao invés do seu `identificador` ou `classe`? Faça o simples e use um seletor de tipo. Se você precisar selecionar todas as listas não ordenadas, use `ul {}`.

[Visualizar Demonstração](#)

Compatibilidade

- IE6+
- Firefox
- Chrome
- Safari
- Opera

6. X:visited e X:link

```
1 | a:link { color: red; }  
2 | a:visited { color: purple; }
```

Nós usamos a pseudo-classe `:link` para selecionar todas as âncoras que foram clicadas.

Alternativamente, nós também temos a pseudo-classe `:visited`, que, como esperado, permite-nos aplicar uma estilização específica somente às âncoras da página que *já foram clicadas* ou *visitadas*.

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

7. X + Y

```
1 | ul + p {  
2 |   color: red;  
3 | }
```

Esse é chamado de seletor adjacente. Ele selecionará *somente* o elemento imediatamente após o primeiro elemento. No exemplo acima, selecionará só o primeiro parágrafo após cada `ul` na página, e fará que ele tenha a cor vermelha.

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

8. X > Y

```
1 | div#container > ul {  
2 |   border: 1px solid black;  
3 | }
```

A diferença entre o seletor descendente `x y` padrão e o seletor `x > y` é que o último somente seleciona filhos diretos. Por exemplo, considere o código HTML abaixo.

```
01 | <div id="container">  
02 |   <ul>  
03 |     <li> Item da lista  
04 |       <ul>  
05 |         <li> Filho </li>  
06 |       </ul>  
07 |     </li>  
08 |     <li> Item da lista </li>  
09 |     <li> Item da lista </li>  
10 |     <li> Item da lista </li>  
11 |   </ul>  
12 | </div>
```

Um seletor `#container > ul` somente selecionará aquelas `ul`s que são filhas diretas de uma `div` com `identificador` nomeado como `container`. Ele não selecionará, por exemplo, a `ul` que é filha do primeiro `li`.

Por esse motivo, há benefícios relacionados a performances ao usar o combinador de filhos. Na verdade, é recomendável utilizá-lo quando estiver trabalhando com motores de seleção de CSS criados em JavaScript.

Visualizar Demonstração

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

9. X ~ Y

```
1 | ul ~ p {  
2 |   color: red;  
3 | }
```

O seletor de irmão é parecido com o seletor `X + Y`, contudo, é menos restritivo. Enquanto o seletor adjacente (`u1 + p`) só selecionará o primeiro elemento, imediatamente, após o elemento inicial, esse é mais generalista. Ele selecionará, usando o nosso exemplo acima, qualquer elemento `p`, desde que ele venha depois de um elemento `u1`.

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

10. X[title]

```
1 a[title] {  
2   color: green;  
3 }
```

Chamado de *seletor de atributos*, no nosso exemplo acima, ele só selecionará aquelas âncoras com um atributo `title`. As âncoras que não tiverem esse atributo, não receberão esse estilo em particular. Mas, e se precisarmos ser mais específicos? Bem...

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

11. X[href="foo"]


```
1 a[href="http://net.tutsplus.com"] {  
2   color: #1f6053; /* nettuts green */  
3 }
```

O trecho acima estilizará todas as âncoras que apontem para o endereço `http://net.tutsplus.com`; eles serão estilizados com o verde do nosso site. Todas as outras âncoras não serão afetadas.

Atente que estamos envolvendo o valor entre aspas. Lembre-se de fazer isso quando estiver trabalhando com motores de seleção CSS criados em JavaScript. Quando possível, sempre use os seletores CSS3 ao invés dos métodos não oficiais.

Ele funciona muito bem, embora seja um tanto rígido. E se o link, realmente, direcionar para o Nettuts+, mas, o endereço usado seja o `nettuts.com` ao invés da URL completa? Para esses casos, podemos usar a sintaxe de expressões regulares.

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

12. X[href*="nettuts"]

```
1 a[href*="tuts"] {  
2   color: #1f6053; /* nettuts green */  
3 }
```

Aqui está; é isso o que precisamos. O asterísco designa que o valor utilizado no seletor deve aparecer em *algum lugar* do valor do atributo do elemento. Assim, esse novo seletor cobre `nettuts.com`, `net.tutsplus.com`, e até o `tutsplus.com`.

Tenha em mente que ele é bem abrangente. E se a âncora apontar para algum site que não seja da Envato, mas que tenha em sua URL a cadeia de caracteres *tuts?* Quando precisar ser mais específico, use `^` e `$`, para referenciar o começo e o fim de uma cadeia de caracteres, respectivamente.

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

13. X[href^="http"]

```
1 a[href^="http"] {  
2   background: url(path/to/external/icon.png) no-repeat;  
3   padding-left: 10px;  
4 }
```

Você, alguma vez, já parou para pensar como alguns sites fazem para mostrar um pequeno ícone ao lado de alguns links que levam para outros sites? Tenho certeza que você já viu algo parecido antes; Eles são ótimos lembretes que o link em questão levará você para um site completamente diferente.

Isso é bem fácil de fazer, usando o acento circunflexo. Ele é, comumente, usado em expressões regulares para designar o começo de uma cadeia de caracteres. Se quisermos selecionar todas as âncoras que tem um atributo `href` que começam com `http`, nós poderíamos usar um trecho de código parecido com o que temos acima.

Atente que não estamos procurando por `http://`; isso é desnecessário e não leva em consideração as URLs que começam com `https://`.

Agora, e se quiséssemos estilizar todos os links que apontam, digamos, para uma foto? Nesses casos, temos de pesquisar pelo *final* da cadeia de caracteres.

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

14. X[href\$=".jpg"]

```
1 a[href$=".jpg"] {  
2   color: red;  
3 }
```

Nós usaremos um símbolo de expressão regular, o `$`, para referenciar o final da cadeia de caracteres. Nesse caso, nós buscaremos por todas as âncoras que apontem para uma imagem – ou, pelo menos, uma URL que termine com `.jpg`. Lembre-se que isso não funcionará com `gifs` e `pngs`.

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

15. X[data-*= "foo"]

```
1 a[data-filetype="image"] {  
2   color: red;  
3 }
```

Lembre-se do exemplo oito; como fazer para referenciar todos os tipos de imagens: `png`, `jpeg`, `jpg` e `gif`? Bem, poderíamos criar múltiplos seletores, dessa forma:

```
1 a[href$=".jpg"],
2 a[href$=".jpeg"],
3 a[href$=".png"],
4 a[href$=".gif"] {
5   color: red;
6 }
```

Mas, isso é muito chato e ineficiente. Outra possível solução é usar atributos customizados. E se adicionarmos um atributo `data-filetype` a cada uma das âncoras que apontem para uma imagem?

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

Então, com esse *gancho*, podemos usar os seletores de atributo padrão para selecionar somente essas âncoras.

```
1 a[data-filetype="image"] {
2   color: red;
3 }
```

Visualizar Demonstração

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

16. X[foo~="bar"]

```
1 a[data-info~="external"] {
2   color: red;
3 }
4
5 a[data-info~="image"] {
6   border: 1px solid black;
7 }
```

Eis um seletor especial que impressionará seus amigos. Nem todo mundo sabe desse truque. O til (`~`) permite-nos selecionar um atributo que tem, em seus valores, uma

lista separada por espaços.

Seguindo com o nosso atributo customizado do seletor quinze, poderíamos criar um atributo `data-info`, que pode receber uma lista de itens, separados por espaço, para podermos anotar o que quisermos. Nesse caso, nós tomaremos nota dos links externos e dos links para imagens – só para exemplificar.

```
1 | "<a href="path/to/image.jpg" data-info="external image"> Clique em mim </a>
```

Com esse código, agora podemos selecionar qualquer tag que tenha qualquer um desses valores, usando o truque do seletor de atributos com o `~`.

```
1 | /* Seleciona o atributo data-info que contem o valor "external" */
2 | a[data-info~="external"] {
3 |   color: red;
4 | }
5 |
6 | /* E aquele que contem o valor "image" */
7 | a[data-info~="image"] {
8 |   border: 1px solid black;
9 | }
```

Bem legal, hein?

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

17. X:checked

```
1 | input[type=radio]:checked {
2 |   border: 1px solid black;
3 | }
```

Essa pseudo-classe somente selecionará o elemento da interface que foi marcado como selecionado – como um botão `radio` ou `checkbox`. Simples assim.

[Visualizar Demonstração](#)

Compatibilidade

- IE9+
- Firefox
- Chrome
- Safari
- Opera

18. X:after

As pseudo-classes `before` e `after` são muito boas. Parece que, todos os dias, o pessoal encontra maneiras criativas e efetivas de usá-las. Elas, simplesmente, geram conteúdo ao redor do elemento selecionado.

Vários aprenderam sobre eles quando foram encontrar o *hack* do ajuste dos *floats* (alguns conhecem como *clear-fix hack*).

```
01 .clearfix:after {
02   content: "";
03   display: block;
04   clear: both;
05   visibility: hidden;
06   font-size: 0;
07   height: 0;
08 }
09
10 .clearfix {
11   *display: inline-block;
12   _height: 1%;
13 }
```

Esse *hack* usa a pseudo-classe `:after` para adicionar um espaço após o elemento e, então, limpá-lo. É um ótimo truque para se ter em seu cinto de utilidades, principalmente quando a técnica do `overflow: hidden;` não é possível ser utilizada.

Para outros usos criativos dessa técnica, [veja meu guia rápido sobre como criar sombras](#).

De acordo com a especificação dos Seletores CSS3, você deveria, tecnicamente, usar a sintaxe dos pseudo-elementos de dois "dois pontos", `::`. Entretanto, para manter-se compatível, o navegador aceita o uso com um único "dois pontos". Na verdade, é melhor você usar a versão com um único "dois pontos" em seus projetos.

Compatibilidade

- IE8+
- Firefox
- Chrome
- Safari
- Opera

19. X:hover

```
1 | div:hover {  
2 |   background: #e3e3e3;  
3 | }
```

Vamos lá. Você já conhece esse. O termo oficial para esse seletor é `pseudo-classe de ação do usuário`. Pode parecer confuso, mas não é. Quer adicionar algum estilo específico ao passar o mouse sobre algum elemento? Esse seletor dará conta do recado!

Tenha em mente que versões antigas do Inter Explorer não respondem quando a pseudo-classe `:hover` é aplicada a qualquer outra coisa que não seja uma âncora (`a`).

Você, geralmente, selecionará esse seletor quando for aplicar, por exemplo, um `border-bottom` a âncoras quando passar o mouse sobre o link.

```
1 | a:hover {  
2 |   border-bottom: 1px solid black;  
3 | }
```

Dica - `border-bottom: 1px solid black;` *é bem melhor que usar*
`text-decoration: underline;`

Compatibilidade

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

20. X:not(selector)

```
1 | div:not(#container) {  
2 |   color: blue;  
3 | }
```

A pseudo-classe `negação` é, particularmente, útil. Digamos que você queira selecionar todas as divs, exceto por aquelas que contenham um `identificador` nomeado de `container`. O trecho acima lida com essa tarefa, perfeitamente.

Ou, se eu quisesse selecionar todos os elementos (o que não é recomendado), exceto pelas tags de parágrafo, poderíamos fazer assim:

```
1 | *:not(p) {  
2 |   color: green;  
3 | }
```

[Visualizar Demonstração](#)

Compatibilidade

- IE9+
- Firefox
- Chrome
- Safari
- Opera

21. X::pseudoElemento

```
1 p::first-line {  
2   font-weight: bold;  
3   font-size: 1.2em;  
4 }
```

Nós podemos usar pseudo-elementos (designados por `::`) para estilizar fragmentos de um elemento, como a primeira linha ou uma primeira letra. Lembre-se que essa regra precisa ser aplicada a elementos do tipo bloco para surtirem efeito.

Um pseudo-elemento é composto por dois "dois pontos":

`::`

Selecione a Primeira Letra de um Parágrafo

```
1 p::first-letter {  
2   float: left;  
3   font-size: 2em;  
4   font-weight: bold;  
5   font-family: cursive;  
6   padding-right: 2px;  
7 }
```

Esse trecho encontrará os parágrafos de uma página e, então, buscará só a primeira letra de cada um desses elementos para aplicar o estilo.

Isso é, geralmente, usado para criar estilos que remetam à estilização que jornais aplicam à primeira letra de um artigo.

Selecione a Primeira Linha de um Parágrafo

```
1 p::first-line {  
2   font-weight: bold;  
3   font-size: 1.2em;  
4 }
```

De forma semelhante, o pseudo-elemento `::first-line` irá, como esperado, estilizar, somente, a primeira linha do elemento em questão.

"Para manter a compatibilidade com as folhas de estilos existentes, os navegadores devem, também, aceitar a notação anterior, com um 'dois pontos', introduzida nas

especificações CSS nível 1 e 2 (precisamente, :first-line, :first-letter, :before e :after). Essa compatibilidade não é permitida para os novos pseudo-elementos nesta especificação." - Fonte

[Visualizar Demonstração](#)

Compatibilidade

- IE6+
- Firefox
- Chrome
- Safari
- Opera

22. X:nth-child(n)

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

Você se lembra quando tinha de indicar um elemento específico de um conjunto de elementos? a pseudo-classe `nth-child` acaba com esse problema!

Atente que o `nth-child` aceita um número inteiro como parâmetro, entretanto, a contagem não inicia em zero (como nas `arrays` e `strings` em programação). Se você deseja selecionar o segundo item de uma lista, faça assim `li:nth-child(2)`.

Nós podemos, até mesmo, selecionar um número variável de elementos filhos. Por exemplo, nós podemos fazer algo como `li:nth-child(4n)`, para selecionar todos os itens que estejam em uma posição que seja múltipla de quatro.

[Visualizar Demonstração](#)

Compatibilidade

- IE9+
- Firefox 3.5+
- Chrome

- Safari

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

```
1 li:nth-last-child(2) {  
2   color: red;  
3 }
```

E se tivéssemos uma enorme lista de itens em uma `ul` e só precisássemos acessar o terceiro item, começando a contagem a partir do último elemento? Ao invés de fazer algo como `li:nth-child(397)`, você pode usar a pseudo-classes `nth-last-child`.

Essa técnica funciona quase igual à técnica dezesseis, entretanto, a diferença é que ela começa do último elemento do conjunto e faz o caminho inverso até o primeiro elemento.

[Visualizar Demonstração](#)

Compatibilidade

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

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

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

Haverá vezes que, ao invés de selecionar um elemento `filho`, você precisará de um certo elemento de um certo `tipo`.

Imagine um código que contenha cinco listas não ordenadas. Se você quiser estilizar, somente, a terceira `ul` e não tem um `identificador` para usar, você pode usar a

pseudo-classe `nth-of-type(n)`. No trecho acima, somente a terceira `ul` terá uma borda.

[Visualizar Demonstração](#)

Compatibilidade

- IE9+
- Firefox 3.5+
- Chrome
- Safari

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

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

E, sim, para manter a consistência, nós também podemos usar a pseudo-classe `nth-last-of-type` para iniciar a contagem dos seletores pelo fim da lista e fazer o caminho inverso até o elemento desejado.

Compatibilidade

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

26. X:first-child

```
1  ul li:first-child {  
2    border-top: none;  
3  }
```

Essa pseudo-classe estrutural permite-nos selecionar o primeiro elemento do conjunto alvo. Você, geralmente, usará essa técnica para remover bordas dos primeiros e últimos elementos de listas.

Por exemplo, digamos que você tem uma lista e cada item dessa lista tem `border-top` e `border-bottom`. Bem, com esse arranjo, o primeiro e último elementos parecerão um pouco estranhos.

Muitos projetistas aplicam classes chamadas `primeira` e `última` para compensar isso. Para evitar esse tipo de coisa, você pode usar essa pseudo-classe.

[Visualizar Demonstração](#)

Compatibilidade

- IE7+
- Firefox
- Chrome
- Safari
- Opera

27. X:last-child

```
1 | ul > li:last-child {  
2 |   color: green;  
3 | }
```

O seletor oposto ao `first-child`, o `last-child`, selecionará o último item do conjunto alvo.

Exemplo

Vamos construir um exemplo simples para demonstrar um dos possíveis usos dessas pseudo-classes. Criaremos uma lista estilizada.

Markup

```
1 | <ul>  
2 |   <li> Item da Lista </li>  
3 |   <li> Item da Lista </li>  
4 |   <li> Item da Lista </li>  
5 | </ul>
```

Nada especial, aqui. Só uma lista simples.

CSS

```
01  ul {  
02    width: 200px;  
03    background: #292929;  
04    color: white;  
05    list-style: none;  
06    padding-left: 0;  
07  }  
08  
09  li {  
10    padding: 10px;  
11    border-bottom: 1px solid black;  
12    border-top: 1px solid #3c3c3c;  
13  }
```

Esse estilo trabalhará o plano de fundo (`background`), removerá o preenchimento padrão (`padding`) que o navegador atribui à `ul` e aplica as bordas a cada elemento `li` para criar um ar de profundidade.

Para criar um ar de profundidade às suas listas, aplique uma cor à `border-bottom` a cada `li`, com um ou dois tons mais escuros que a cor do plano de fundo do `li`. Depois, aplique uma cor à `border-top` que seja um ou dois tons mais claros.

O único problema, como mostrado na imagem acima, é que uma borda será aplicada à parte de cima e à parte de baixo da lista desordenada – o que é estranho. Vamos usar as pseudo-classes `:first-child` e `:last-child` para ajustar isso.

```
1  li:first-child {  
2    border-top: none;  
3  }  
4  
5  li:last-child {  
6    border-bottom: none;  
7  }
```

E é isso. Resolvemos!

[Visualizar Demonstração](#)

Compatibilidade

- IE9+
- Firefox
- Chrome
- Safari
- Opera

SIM – o IE8 dá suporte ao `:first-child`, mas não ao `:last-child`. Vai entender.

28. X:only-child

```
1 | div p:only-child {  
2 |   color: red;  
3 | }
```

Francamente, você, provavelmente, não usará muito a pseudo-classe `only-child`. De qualquer modo, ela existe para o caso de você precisar.

Ela permite que você selecione aqueles elementos que são os *únicos* filhos de um certo elemento. Levando em consideração o trecho acima, somente o parágrafo que for o único filho de uma `div` ficará colorido de vermelho.

Vamos assumir que seja esse o nosso HTML.

```
1 | <div><p> Meu parágrafo aqui. </p></div>  
2 |  
3 | <div>  
4 |   <p> Dois parágrafos no total. </p>  
5 |   <p> Dois parágrafos no total. </p>  
6 | </div>
```

Nesse caso, os parágrafos da segunda `div` não serão afetados; somente os da primeira `div`. Assim que você inserir mais de um elemento filho nessa `div`, a pseudo-classe `only-child` deixará de funcionar.

[Visualizar Demonstração](#)

Compatibilidade

- IE9+
- Firefox
- Chrome
- Safari
- Opera

29. X:only-of-type

```
1 li:only-of-type {  
2   font-weight: bold;  
3 }
```

Essa pseudo-classe estrutural pode ser usada de algumas formas bem interessantes. Ela selecionará elementos que não tem nenhum elemento irmão dentro do elemento pai. Como exemplo, selecionemos todas as `ul`s que tem somente um único `li`.

Primeiro, pergunte-se como você resolveria isso. Você poderia usar o `ul li`, mas, ele selecionaria *todos* os itens da lista. A única solução é usar a pseudo-classe `only-of-type`.

```
1 ul > li:only-of-type {  
2   font-weight: bold;  
3 }
```

Visualizar Demonstração

Compatibilidade

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

30. X:first-of-type

A pseudo-classesse `first-of-type` permite que você selecione o primeiro irmão de um conjunto de elementos do mesmo tipo.

Um Teste

Para entender melhor, façamos um teste. Copie o código abaixo em seu editor de códigos:

```
01 <div>
02   <p> Meu parágrafo aqui. </p>
03   <ul>
04     <li> Item 1 da Lista </li>
05     <li> Item 2 da Lista </li>
06   </ul>
07
08   <ul>
09     <li> Item 3 da Lista </li>
10     <li> Item 4 da Lista </li>
11   </ul>
12 </div>
```

Agora, sem continuar lendo, descubra como selecionar somente o *"Item 2 da Lista"*. Quando descobrir (ou desistir), continue a leitura.

Solução 1

Há várias maneira de resolver esse teste. Nós usaremos alguns deles. Primeiro, usando o `first-of-type`.

```
1  ul:first-of-type > li:nth-child(2) {
2    font-weight: bold;
3  }
```

Esse trecho de código, essencialmente, diz "encontre a primeira lista não ordenada na página, então encontre somente os filhos imediatos dessa lista e que são "itens de lista". Depois, filtre esses itens e pegue somente o segundo.

Solução 2

Outra opção seria usar o seletor adjacente.

```
1  p + ul li:last-child {
2    font-weight: bold;
3  }
```

Nesse cenário, nós encontramos a `ul` que está, imediatamente, após um `p`, e, então, encontramos o último item desse elemento.

Solução 3

Nós podemos ser bem desagradáveis ou brincalhões o quanto quisermos com esses seletores.

```
1  ul:first-of-type li:nth-last-child(1) {  
2    font-weight: bold;  
3  }
```

Dessa vez, nós pegamos o primeiro elemento `ul` da página e, então, encontramos o primeiro item da lista, na ordem reversa dos itens! :)

Visualizar Demonstração

Compatibilidade

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

Conclusão

Se você tem de lidar com navegadores antigos, como o Internet Explorer 6, você ainda tem de tomar cuidado com esses novos seletores. Porém, não deixe isso deter seu aprendizado. Você estaria fazendo um desserviço a você mesmo. Tenha certeza [de visualizar a lista de compatibilidade com os navegadores](#). Alternativamente, você pode usar [o excelente script IE9.js do Dean Edward](#), para permitir que esses seletores sejam usados em navegadores.

Segundo, quando for trabalhar com bibliotecas, como a popular jQuery, sempre use os seletores nativos do CSS3 ao invés dos métodos/seletores customizados da biblioteca, sempre que possível. Isso [fará seu código mais performático](#), uma vez que o motor de seleção poderão usar o analisador nativo do navegador, ao invés do seu próprio.

Obrigado por ler! Espero que tenha aprendido uma ou dois truques!

Seja o primeiro a saber sobre novas traduções—siga [@tutsplus_pt](#) no Twitter!

Advertisement



Jeffrey Way

I used to be the editor of Nettuts+ and head of web development courses at Tuts+.

[🐦 jeffrey_way](#)

[📡 FEED](#) [👍 LIKE](#) [🐦 FOLLOW](#) [👤 FOLLOW](#)

Weekly email summary

Subscribe below and we'll send you a weekly email summary of all new Code tutorials. Never miss out on learning about the next big thing.

Update me weekly

Advertisement

Translations


Envato Tuts+ tutorials are translated into other languages by our community members—you can be involved too!

Translate this post

Powered by  native

12 COMENTÁRIOS

Nettuts+

 Iniciar sessão ▾ Recomendar 4 Tweet Partilhar

Mostrar primeiro os mais votados ▾



INICIE SESSÃO COM O

OU REGISTE-SE NO DISQUS **Murilo Giachini Ferro** • há 2 anos

Show, corri para o js em busca de uma solução com expressão regular resolver meu problema, mas não precisei quando achei aqui o truque x:nth-child(numero)
Valew querido, sucesso! Viva o CSS!

1 ^ | ▾ • Responder • Partilhar ›

**Lucas Medina** • há 3 anos

Excelentes dicas :) alguns seletores muito compatíveis, como os de expressão regular, são muito úteis e eu nunca utilizei deles! Vale super a pena. Obrigado pelas dicas ;)

1 ^ | ▾ • Responder • Partilhar ›

**Irmão Ton** • há 3 anos

Muito valioso suas dicas!

1 ^ | ▾ • Responder • Partilhar ›

**Lineu Gomes Azevedo** • há 2 anos

Ótimo artigo, O item com o seletor X:first-of-type apresenta 12 soluções diferentes.

^ | ▾ • Responder • Partilhar ›

**Amandita** • há 2 anos

Gostei parabéns pelo artigo

^ | ▾ • Responder • Partilhar ›

**Armando Ricardo** • há 2 anos

Como estilizar somente os primeiros elementos LI dos elementos UL filhos direto de #CONTAINER ?

Segue estrutura:

```
<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>
```

[ver mais](#)

^ | v • Responder • Partilhar ›



Lineu Gomes Azevedo ➔ Armando Ricardo • há 2 anos

Inclua uma tag span onde quer estilizar

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

E agora é só inserir um seletor interno no código CSS:

```
li span{
  color:red;
}
```

1 ^ | v • Responder • Partilhar ›



William ➔ Armando Ricardo • há 2 anos

```
#container > ul > li:first-child {...}
```

^ | v • Responder • Partilhar ›



Armando Ricardo ➔ William • há 2 anos



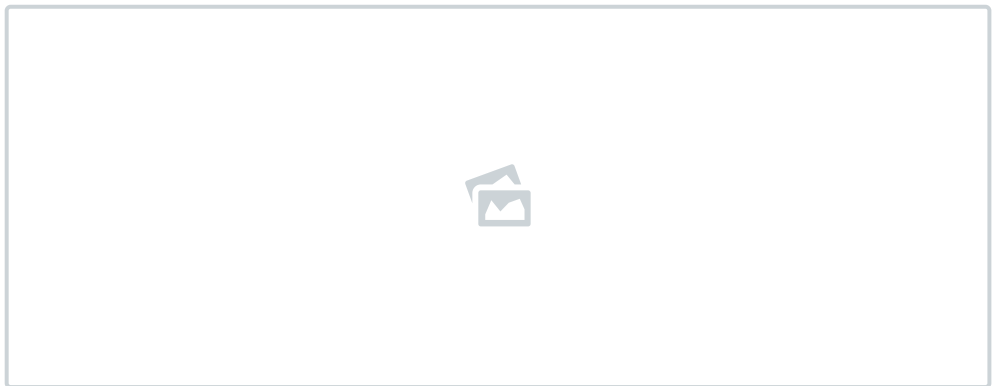
Já tentei assim e resultado não é satisfatório...

Somente os primeiros deveriam estar em vermelho (não incluindo os filhos de LI)

^ | v • Responder • Partilhar ›



Armando Ricardo ➔ Armando Ricardo • há 2 anos



Eu queria que ficasse assim:

^ | v • Responder • Partilhar ›



Erick Patrick ➔ Armando Ricardo • há 2 anos

Olá Armando,

Acredito que o que você busca é isso:

```
#container ul li:first-child {  
  color: red  
}
```

```
#container ul li:first-child ul li {  
  color: black  
}
```

Para ver um exemplo real, veja só [essa CodePen](#).

O ideal, porém, era que você não usasse seletores tão aninhados e passasse a usar alguma classe para estilizar os elementos que você quer.

1 ^ | v • Responder • Partilhar ›



William ➔ Armando Ricardo • há 2 anos

Eu trocaria as por <dl>, as que quer modificar por <dt> e o resto por <dd>. Aí aplica as modificações nas <dt>.

^ | v • Responder • Partilhar ›

Advertisement

QUICK LINKS - Explore popular categories

ENVATO TUTORIALS



JOIN OUR COMMUNITY



[HELP](#)

tuts+

27,328

Tutorials

1,220

Courses

38,331

Translations

[Envato.com](#) [Our products](#) [Careers](#) [Sitemap](#)

© 2019 Envato Pty Ltd. Trademarks and brands are the property of their respective owners.

[Follow Envato Tuts+](#)