Pseudo-Classes do CSS

> As Pseudo-Classes do CSS são palavras adicionadas a um seletor, definindo um estado especial do elemento em questão.

Vamos imaginar que temos um botão na nossa página e queremos que, ao clicar nesse botão, ele mude sua cor de vermelho para verde.

Normalmente, se não conhecêssemos as pseudo-classes, faríamos uso de JavaScript mudando a cor do background para verde e ao fim do click retornando a cor de fundo para vermelho, correto? Sim, está correto, todavia, existe um jeito muito mais simples de fazer esse tipo de operação, vamos ver a seguir alguns métodos que existem dentro de CSS que satisfazem o que queremos executar.

🎨 **Pseudo-Classes:**

Aa Nome	Exemplo de uso	O que faz
:active	`a:active { }`	Seleciona o link quando ativo.
:checked	`input[type="checkbox"]:checked { }`	Seleciona o elemento quando checado (option e inputs do tipo radio e tipo checkbox).
:disabled	`input:disabled { }`	Seleciona o elemento quando está desabilitado.
:empty	`div:empty { }`	Seleciona o elemento que não possui nenhum filho.
:enable	`input:enabled { }`	Seleciona o elemento quando está habilitado.
:first-child	`p:first-child { }`	Seleciona o primeiro elemento de um grupo de elementos irmãos.
:first-of-type	`p:first-of-type { }`	Seleciona o primeiro elemento de seu tipo entre um grupo de elementos irmãos.
:focus	`input:focus { }`	Seleciona o elemento quando recebe o foco.
:hover	`h1:hover { }`	Seleciona o elemento que passamos o mouse por cima.
:in-range	`input:in-range { }`	Seleciona o elemento cujo valor está dentro dos limites especificados pelos atributos `min` e `max` do input.
:invalid	`input:invalid { }`	Seleciona o elemento cujo conteúdo não seja válido.
:lang()	`h2:lang(idioma) { }`	Seleciona o elemento com base no idioma em que eles estão determinados.
:last-child	`h2:last-child { }`	Seleciona o último elemento de um grupo de elementos irmãos.
:last-of-type	`h2:last-of-type { }`	Seleciona o último elemento de seu tipo entre um grupo de elementos irmãos.
:link	`a:link { }`	Seleciona os links que ainda não foram visitados.
:not()	`:not(elemento) { }`	Seleciona os elementos que não sejam iguais ao elemento passado entre parênteses.
:nth-child(n)	`li:nth-child(posicao) { }`	Seleciona os elementos irmãos a partir da poção passada. Dois casos de uso: `odd` → posições ímpares / `even` → posições pares.
:nth-last-child(n)	`li:nth-last-child(posicao) { }`	Seleciona o último elemento entre o grupo de irmãos. A contagem se inicia do último e vai em direção ao primeiro.
:nth-last-of-type(n)	`li:nth-last-of-type(posicao) { }`	Seleciona elementos de determinado tipo a partir do grupo de irmãos. A contagem também se inicia do último e vai em direção ao primeiro.
:nth-of-type(n)	`li:nth-of-type(posicao) { }`	Seleciona o elemento com base no seu tipo e na sua posição no grupo de irmãos.
:only-child	`span:only-child {}`	Seleciona o elemento que é o único elemento filho de seu elemento pai.
:optional	`input:optional { }`	Seleciona os elementos <input> que não possuem o atributo required.
:out-of-range	`input:out-of-range { }`	Seleciona o elemento cujo valor está fora dos limites especificados pelos atributos `min` e `max` do input.
:read-only	`p:read-only { }`	Seleciona o elemento que não pode ser editado pelo usuário. Caso o elemento pertencente ao grupo de seus irmãos não possua o atributo `contenteditable`.

Aa Nome	Exemplo de uso	O que faz
:read-write	`p:read-write { }`	Seleciona o elemento que pode ser editado pelo usuário. Ou seja, seleciona o elemento que possui o atributo `contenteditable`.
:required	`input:required { }`	Seleciona os elementos que possuem o atributo `required`.
:root	`:root { }`	Corresponde ao elemento raíz de uma árvore que representa o documento.
:target	`:target { }`	Representa um elemento único com um fragmento correspondente a URL. Por exemplo: http://www.exemplo.com/index.html#elemento2, o elemento com id = `elemento2` seria selecionado.
:valid	`input:valid { }`	Seleciona o elemento cujo conteúdo seja válido.
:visited	`a:visited { }`	Seleciona todos os links já visitados.

Possuímos bastante possibilidades não é mesmo? Agora separamos algumas mais usadas e alguns exemplos para treino:

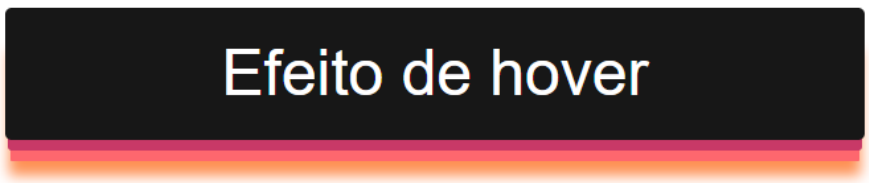
- :hover →
  - Usamos o efeito de hover para adicionar estilos ao usuário passar o cursor do mouse por cima do elemento. Por exemplo:
  - HTML:

```
<body>
  <button class="btn">
    Efeito de hover
  </button>
</body>
```

  - CSS:

```
/* Zeramos os espaçamentos padrões do navegador: */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

/* Estilo do corpo do HTML */
body {
  height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
}

/* Estilo padrão do botão: */
.btn {
  width: 650px;
  height: 100px;
  font-size: 3em;
  cursor: pointer;
  background-color: #171717;
  color: white;
  border: none;
  transition: all .4s;
}

/* Estilo para o efeito de hover: */
.btn:hover {
  border-radius: 5px;
  transform: translate(-10px);
  box-shadow: 0 10px 0 -2px #C83967,
  0 20px 0 -4px #FE676E,
  0 30px 10px -3px #FE9053;
}

/* Estilo para o efeito de active: */
.btn:active {
  background-color: #8f2e4d;
  transition: all 0.1s ease;
}
```

O resultado é igual a:

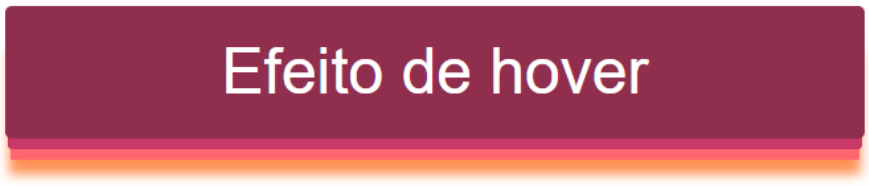


- :active→
  - Usamos esse efeito quando ativamos o elemento em questão, como em um clique. Por exemplo:
  - HTML:

```
<body>
  <button class="btn">
    Efeito de hover
  </button>
</body>
```

  - CSS:

```
/* Zeramos os espaçamentos padrões do navegador: */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

/* Estilo do corpo do HTML */
body {
  height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
}

/* Estilo padrão do botão: */
.btn {
  width: 150px;
  height: 40px;
  font-size: 1.1em;
  cursor: pointer;
  background-color: #171717;
  color: white;
  border: none;
  transition: all .4s;
}

/* Estilo para o efeito de hover: */
.btn:hover {
  border-radius: 5px;
  transform: translate(-10px);
  box-shadow: 0 7px 0 -2px #C83967,
  0 15px 0 -4px #FE676E,
  0 16px 10px -3px #FE9053;
}

/* Estilo para o efeito de active: */
.btn:active {
  background-color: #8f2e4d;
  transition: all 0.1s ease;
}
```



Perceba que utilizamos a mesma estrutura que o exemplo da pseudo-classe anterior, apenas adicionamos o efeito para o efeito quando ativado.

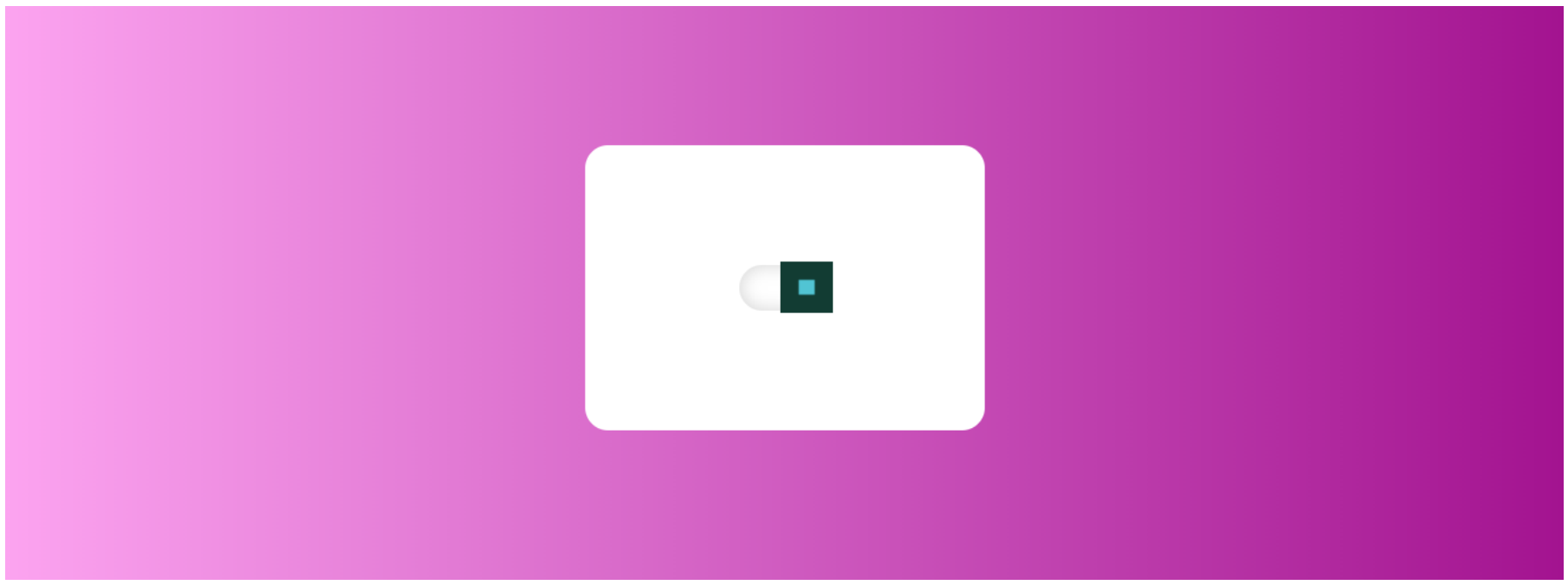
- :checked →
  - Esse efeito é usado para quando "checamos" um input de tipo radio ou checkbox. O seguinte exemplo é um pouco extenso, porém aborda bastante tópicos de estilização:
  - HTML:

```
<body>
  <div class="main">
    <div class="container">
      <input
        type="checkbox"
        name="toggle"
        id="toggle">
    </div>
  </div>
</body>
```

  - CSS:

```
/* Zeramos os espaçamentos padrões do navegador: */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

/* Estilo do corpo do HTML */
body {
  height: 100vh;
  display: flex;
  justify-content: center;
  align-items: center;
  background: linear-gradient(to right, #fca4f0, #a31390);
}

/* Estilo da classe main: */
.main {
  width: 350px;
  height: 250px;
  background-color: #fff;
  display: flex;
  justify-content: center;
  align-items: center;
  border-radius: 20px;
}

/* Estilo da classe container: */
.container {
  width: 80px;
  height: 40px;
  box-shadow: inset 0 0 15px rgba(0, 0, 0, 0.171);
  border-radius: 20px;
}

/* Estilos do input dentro da classe container: */
.container input {
  width: 100%;
  height: 100%;
  cursor: pointer;
  appearance: none;
```

```
  -webkit-appearance: none;
  position: relative;
}

.container input::before,
.container input::after {
  content: '';
  position: absolute;
  top: 48%;
  width: 2%;
  height: 4%;
  background: #123c33;
  border-radius: 20px;
  transition: all .2s;
}

.container input::before {
  left: 25%;
  box-shadow: 0 0 0 22px #123c33;
}

.container input::after {
  right: 25%;
  opacity: 0;
}

.container input:checked::before {
  box-shadow: 0 0 0 80px #123c33;
  opacity: 0;
}

.container input:checked::after {
  background-color: #51c4d3;
  box-shadow: 0 0 1px 6px #51c4d3,
  0 0 0 22px #123c33;
  opacity: 1;
}
```

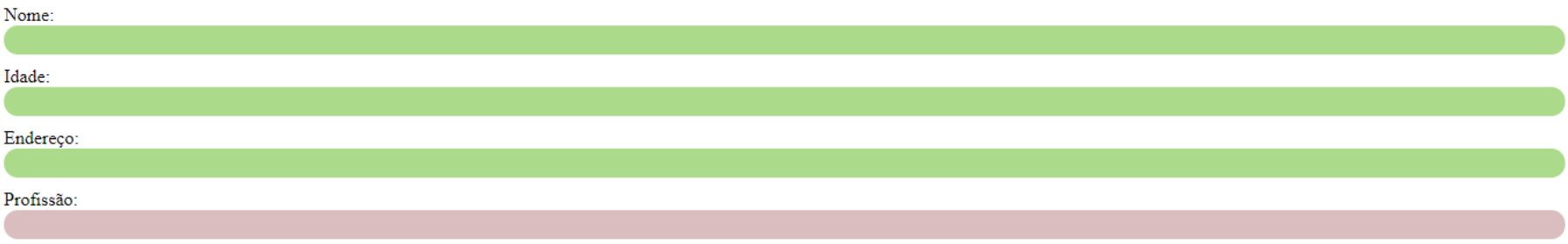


- :disabled →
  - Efeito usado para quando o elemento está desabilitado. Normalmente usamos estilos mais "apagados", indicando que o campo está desativado para preenchimento: Por exemplo:
  - HTML:

```
<body>
  <main>
    <div>
      <label for="nome">Nome:</label>
      <input type="text" name="nome" id="nome">
      <label for="idade">Idade:</label>
      <input type="number" name="idade" id="idade">
      <label for="endereco">Endereço:</label>
      <input type="text" name="endereco" id="endereco">
      <label for="profissao">Profissão:</label>
      <input type="text" name="profissao" id="profissao" disabled>
    </div>
  </main>
</body>
```

- CSS:

```
/* Estilização da div: */
div {
  display: flex;
  flex-direction: column;
}

/* Estilização do input: */
input {
  background-color: #abdb8a;
  border-radius: 15px;
  padding: 5px;
  outline: none;
  border: none;
  margin-bottom: 10px;
}

/* Estilização do input desabilitado: */
input:disabled {
  background-color: #dabdbf;
}
```



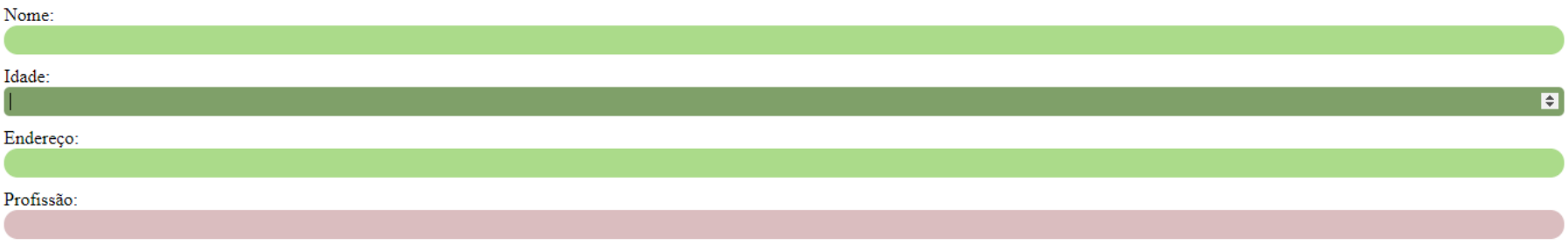
- :focus →
  - Esse efeito é usado para atribuir um efeito ao elemento quando possuir o foco da aplicação. Por exemplo:
  - HTML:

```
<body>
  <main>
    <div>
      <label for="nome">Nome:</label>
      <input type="text" name="nome" id="nome">
      <label for="idade">Idade:</label>
      <input type="number" name="idade" id="idade">
      <label for="endereco">Endereço:</label>
      <input type="text" name="endereco" id="endereco">
      <label for="profissao">Profissão:</label>
      <input type="text" name="profissao" id="profissao" disabled>
    </div>
  </main>
</body>
```

  - CSS:

```
div {
  display: flex;
  flex-direction: column;
}

input {
  background-color: #abdb8a;
  border-radius: 15px;
  padding: 5px;
  outline: none;
  border: none;
  margin-bottom: 10px;
  transition: all 1s ease;
}

input:focus {
  background-color: #93bb79;
  border-radius: 5px;
  transition: all 1s ease;
}
```

```
input:disabled {
  background-color: #dabdbf;
}
```



- :nth-child(n) →
  - Usamos esse efeito para selecionar os elementos irmãos a partir da posição passada entre parênteses. Os dois valores mais comuns para uso além dos números é `odd` → para posições ímpares e `even` → para posições pares. Vamos ver a seguir um exemplo aplicado à uma tabela:
  - HTML:

```
<table id="table">
  <caption>
    Candidatos para o Beginners PRO
  </caption>
  <tr>
    <th>Nome</th>
    <th>Idade</th>
    <th>Profissão</th>
  </tr>
  <tr>
    <td>Clóvis</td>
    <td>30</td>
    <td>Professor</td>
  </tr>
  <tr>
    <td>Silvio Santos</td>
    <td>90</td>
    <td>Apresentador</td>
  </tr>
  <tr>
    <td>Agostinho Carrara</td>
    <td>53</td>
    <td>Taxista</td>
  </tr>
</table>
```

  - CSS:

```
#table{
  text-align: center;
  border: 2px solid #a4a4ff;
  margin: auto;
  width: 40%;
}

#table caption {
  padding: 1vh;
  margin-bottom: 1vh;
  border-radius: 10px;
  background-color: #00A1D7;
  color: white;
}

td, th {
  padding: 1vh;
}

table, th, td{
  border: 1px solid lightgrey;
  border-collapse: collapse;
}

tr:nth-child(odd) {
  background-color: #5BB9D3;
```

```
}

tr:nth-child(even) {
  background-color: #70C2E8;
}

tr:nth-child(1) {
  background-color: #3B70A2;
  color: white;
}

#divTable {
  display: flex;
  justify-content: center;
}
```

Candidatos para o Beginners PRO

Nome	Idade	Profissão
Clóvis	30	Professor
Silvio Santos	90	Apresentador
Agostinho Carrara	53	Taxista

- :valid e :invalid →
  - Esses efeitos são responsáveis por inserir estilos a elementos válidos e inválidos, respectivamente. Podemos utilizá-os separadamente como em conjunto, por exemplo:
  - HTML:

```
<div>
  <label for="nome">Nome:</label>
  <input type="text" name="nome" id="nome">
  <label for="email">Email:</label>
  <input type="email" name="email" id="email">
  <label for="idade">Idade:</label>
  <input type="number" max="10" min="3" name="idade" id="idade">
</div>
```

  - CSS:

```
div {
  display: flex;
  flex-direction: column;
}

input {
  background-color: #abdb8a;
  border-radius: 15px;
  padding: 5px;
  outline: none;
  border: none;
  margin-bottom: 10px;
  transition: all 1s ease;
}

input:valid {
  background-color: #7fa069;
}

input:invalid {
  background-color: #d8333e;
}
```

Nome:
Clóvis
Email:
a@a.com
Idade:
7

Ps.: Sempre recomendamos que você veja os exemplos do material e treine adicionando e/ou mudando os estilos e a estrutura em sua aplicação.

Pseudo-elementos

Assim como pseudo-classes, possuímos os pseudo-elementos. Eles permitem que estilizemos uma parte específica do elemento. Algumas possibilidades são:

- ::after → Insere algo após o conteúdo de cada elemento selecionado.

Por exemplo:

```
<p>O texto será adicionado logo após esse parágrafo: </p>
```

```
p::after {
  content: "oi";
  color: red;
}
```

- ::before → Insere algo antes do conteúdo do elemento selecionado.

Por exemplo:

```
<p>o texto foi adicionado anteriormente pelo ::before</p>
```

```
p::before {
  content: "Oi, ";
  color: red;
}
```

- ::first-letter → Insere um estilo apenas para a primeira letra do seletor especificado.

Por exemplo:

```
  <p>Essa frase começou com uma letra azul</p>
```

```
p::first-letter {
  color: blue;
  font-size: 2em;
}
```

- ::first-line → Insere um estilo apenas para a primeira linha do seletor especificado.

Por exemplo:

```
<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Ad ipsam mollitia, commodi exercitationem, optio
  suscipit ullam aliquam, aperiam iste nihil ratione tenetur a debitis laboriosam accusantium. Iure sequi est ex!
  Ex beatae quas iste unde eum vitae, deleniti nemo dolorum tempore nam repellendus iure. Soluta, omnis quaerat
  veniam reprehenderit adipisci, pariatur enim maxime consequuntur placeat facilis nostrum amet inventore ipsum.
</p>
```

```
p::first-line {
  color: green;
}
```

- ::selection → Estiliza a parte de um elemento que é selecionado pelo usuário.

Por exemplo:

```
<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Ad ipsam mollitia, commodi exercitationem, optio
  suscipit ullam aliquam, aperiam iste nihil ratione tenetur a debitis laboriosam accusantium. Iure sequi est ex!
  Ex beatae quas iste unde eum vitae, deleniti nemo dolorum tempore nam repellendus iure. Soluta, omnis quaerat
  veniam reprehenderit adipisci, pariatur enim maxime consequuntur placeat facilis nostrum amet inventore ipsum.
</p>
```

```
p::selection {
  background-color: red;
  color: white;
}
```

Bibliografias:

- https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes;
- https://www.w3schools.com/css/css_pseudo_classes.asp;