

# Pseudo-seletores I



**Certified  
Developer**  
The Ultimate Tech Degree

**DigitalHouse** >  
Coding School

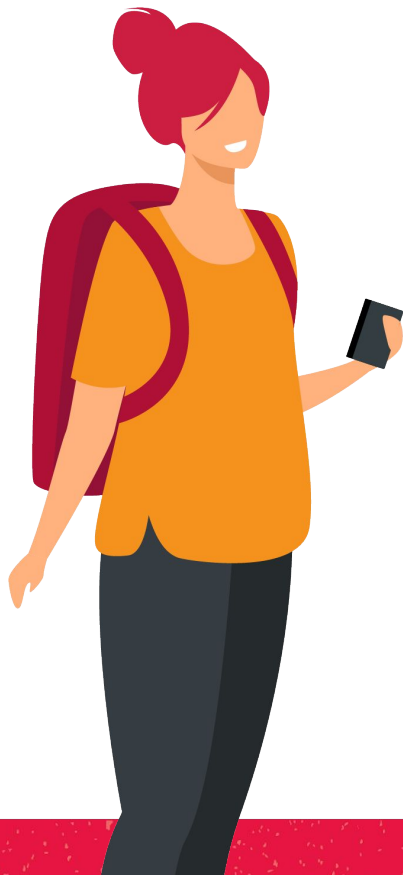


## Temas da aula:

- 1** Pseudo-seletores
- 2** Pseudo-classes
- 3** Pseudo-elementos

**1**

# Pseudo-seletores



“

Os **pseudo-seletores** nos permitem uma maneira alternativa de criar elementos e de aplicar estilos em função dos estados e da organização desses elementos.

”



## Os Pseudo-seletores

Estão divididos em dois grandes grupos:

As **pseudo-classes**, que nos permitem aplicar estilos em função de:

- Os estados dos elementos.
- A publicação dentro da estrutura do HTML.
- A presença de certos atributos dentro do HTML.

Os **pseudo-elementos**, que nos permitem inserir conteúdo pelo CSS sem ter que modificar a estrutura do HTML.

## 2 | Pseudo-classes



## As Pseudo-classes

As pseudo-classes são agregadas nos seletores que já conhecemos.

Normalmente, escrevemos o nome do seletor primeiro, seguido de dois pontos `:` e a pseudo-classe que queremos utilizar.

Algumas pseudo-classes podem se aplicar a todos os elementos HTML, exceto algumas pseudo-classes específicas que só se aplicam a alguns elementos.

css

```
selector:pseudo-classe {  
    propriedade: valor;  
}
```



## As Pseudo-classes

Os links são elementos que tem pseudo-classes específicas.

Neste caso são `:link`, `:visited`, `:hover` e `:active`

CSS

```
a:link,  
a:visited {  
    color: rgb(98, 8, 194);  
    font-weight: bold;  
}  
  
a:hover,  
a:active {  
    color: rgb(136, 11, 74);  
}
```





## As Pseudo-classes (:link)

Se utiliza para **aplicar estilo** aos **links** `<a></a>` que tenham a propriedade `href`.

Os estilos definidos pela pseudo-classe `:link` serão anulados por qualquer pseudo-classe posterior relacionada com esse tipo de link (`:visited`, `:hover` ou `:active`) que tenha ao menos a mesma especificidade.

css

```
a:link {  
    background-color: rgb(234, 0, 255);  
    border-color: rgb(161, 17, 89);  
    color: red;  
}
```



## As Pseudo-classes (:visited)

Se utiliza para **aplicar estilo** aos **links** `<a></a>` que já tenham sido **visitados** pelo menos uma vez pelo usuário.

Seu estilo também pode ser anulado por outras propriedades de tipo de link que foram atribuídas.

css

```
a:visited {  
    background-color: rgb(234, 0, 255);  
    border-color: rgb(161, 17, 89);  
    color: red;  
}
```



## As Pseudo-classes (:hover)

Essa pseudo-classe está relacionada na maioria das vezes com os links, **mas pode ser relacionada com qualquer elemento HTML.**

Se utiliza para **aplicar estilo** a qualquer elemento que o usuário posicione o cursor.

Caso seja aplicada a um link, os estilos que foram atribuídos a ele podem ser anulados por essa pseudo-classe.

css

```
a:hover {  
    background-color: gold;  
}
```



## As Pseudo-classes (:active)

Se utiliza para **aplicar estilo** aos **links** `<a></a>` que estão sendo clicados pelo usuário. Normalmente, se utiliza para fazer uma animação do clique.

Seus estilos também poderão ser anulados por essa pseudo-classe atribuída.

```
css
a:active {
    background-color: rgb(234, 0, 255);
    border-color: rgb(161, 17, 89);
    color: red;
}
```



## As Pseudo-classes dos Inputs

Os **inputs** são outro tipo de elemento que tem pseudo-classes específicas.

Nesse caso, são: `:focus`, `:enabled`, `:disabled` e `:target`

```
css  input:focus { border-color: orange; }  
  
     input:disabled { background-color: gray; }  
  
     :target { font-weight: bold; }
```



## As Pseudo-classes dos Inputs (:focus)

Se aplica quando um elemento **recebe o foco, ou seja, quando o usuário clica nesse elemento.**

O caso mais comum dessa pseudo-classes é em campos de preenchimento: quando o usuário clica, o campo recebe um destaque.

CSS

```
input:focus {  
  color: orange;  
  font-weight: bold;  
}
```



## As Pseudo-classes dos Inputs (:disabled)

Se aplica quando um elemento pode ser considerado como **desabilitado**.

Normalmente, essa pseudo-classe é utilizada em campos no formulário que não podem ser preenchidos por algum motivo.

```
css  input:disabled {  
      background-color: gray;  
  }
```



## Quantas pseudo-classes existem?

Existem diversas pseudo-classes para serem utilizadas. Para descobrir mais pseudo-classes e aprender as peculiaridades sobre cada uma, visite o link: [pseudo-classes](#)



# 3 | Pseudo-elementos



## Os Pseudo-elementos

Os pseudo-elementos também são usados com os seletores CSS.

Para usá-los, escrevemos o nome do seletor primeiro, seguido de dois pontos colocados duas vezes `::` e o pseudo-elemento que queremos utilizar.

Os mais utilizados são `::before` e `::after`, mas existem muitos mais, como: `::first-letter` e `::first-line`.

css

```
selector::pseudo-elemento {  
  propriedade: valor;  
}
```



## Os Pseudo-elementos (::before)

Se utiliza junto com a propriedade `content` para introduzir conteúdo no documento usando CSS.

Esse novo conteúdo aparecerá **antes** do conteúdo interno do elemento.

CSS

```
div::before {  
  content: "Esse aparecerá antes do conteúdo do elemento";  
  color: red;  
}
```



## Os Pseudo-elementos (::after)

Se utiliza junto com a propriedade `content` para introduzir conteúdo no documento usando CSS.

Esse novo conteúdo aparecerá **depois** do conteúdo interno do elemento.

CSS

```
div::after {  
  content: "Esse aparecerá depois do conteúdo do elemento";  
  color: red;  
}
```

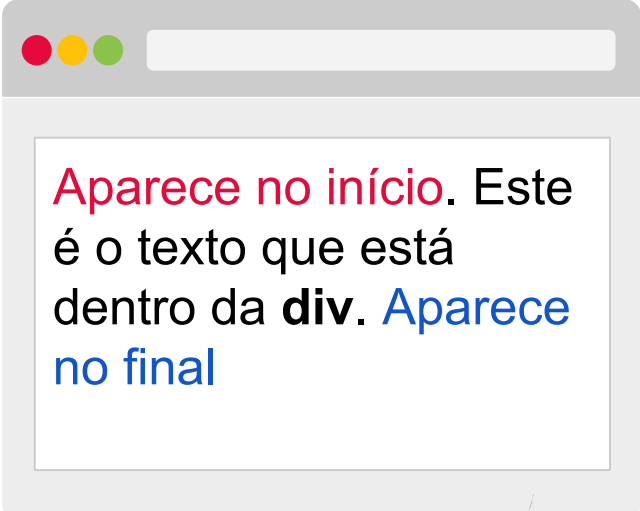


## Como aparece no navegador:

Por padrão `::before` e `::after` são elementos de linha porque aparecerão antes ou depois do conteúdo já existente. Porém, podemos transformá-los em elementos de bloco e usá-los para qualquer coisa que precisemos.

CSS

```
div::before {  
  content: "Aparece no início";  
  color: red;  
}  
  
div::after {  
  content: "Aparece no final";  
  color: blue;  
}
```



Aparece no início. Este  
é o texto que está  
dentro da **div**. Aparece  
no final



## Esquema geral da sintaxe do CSS

Considerando tudo o que aprendemos, a ordem para inserir uma regra dentro do CSS será a seguinte:

CSS

```
selector #id .class :pseudoclasse ::pseudoelemento [atributo]
{
    propriedade: valor;
    propriedade: valor;
    propriedade: valor;
}
```

DigitalHouse>  
Coding School