



# Front-end Essencial CSS

Prof. Yuri Weilemann

# O que é CSS?



PROCURA-SE  
DESIGNER!

CSS significa Cascading Style Sheet. É um código para criar estilos no HTML. Enquanto o HTML é a estrutura, o CSS é a beleza.

Assim como o HTML, o CSS não é uma linguagem de programação, mas sim uma linguagem style sheet, enquanto o HTML é uma linguagem de marcação.

# Apresentando o CodePen.io

O CodePen.io é uma ferramenta online que possibilita a criação de aplicações "completas" utilizando HTML, CSS e JavaScript. Vamos dar uma olhada em um exemplo básico de como o CSS funciona junto do HTML...

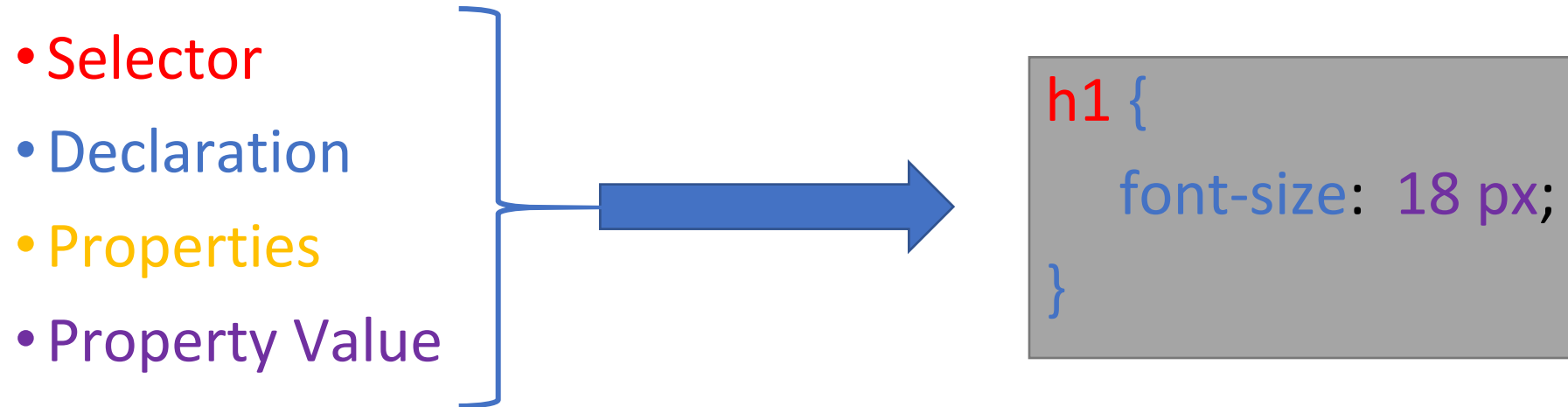


# Comentários

Comentários em blocos podem ser adicionados no CSS através do uso dos símbolos `/* */`. Assim como no HTML, os comentários não afetarão seu código. Geralmente, são utilizados para definir a qual elemento HTML aquelas configurações de estilo estão vinculadas. Por exemplo:

```
/*  
|  Estilização do Container principal da página Home  
*/  
  
h1 {  
    font-size: 18 px;  
    color: blue;  
    font-family: 'Courier New', Courier, monospace;  
}  
  
p {  
    color: red;  
}  
  
#container {  
    width: 300px;  
    height: 200px;  
    background-color: aliceblue;  
}
```

# Anatomia do CSS



# Seletores

Seletores são responsáveis por conectar o CSS com o HTML. Podemos declarar mais de um seletor por bloco de CSS através do uso de vírgulas. Temos alguns tipos de seletores importantes:

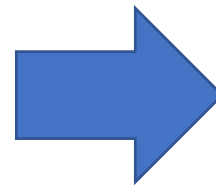
- Global Selector – ' \* '
- Element/Type Selector – h1, h2, p, div...
- ID Selector – #box, #container...
- Class Selector – .title, .subtitles, .description...
- Attribute Selector, Pseudo-Class, Pseudo-Element, entre outros.

# Box Model

Com o tempo, vamos perceber que o CSS trabalha com a ideia de caixas, ou seja, box model. Toda caixa em altura, largura, espaço dentro, conteúdo, etc. Vamos verificar os seguintes códigos trabalhando juntos:

## HTML

```
<h1>Seja mais saudável!</h1>  
<p>Descrição</p>  
<button>Iniciar Treino</button>
```



## CSS

```
h1 {  
  border: 1px solid red;  
  margin: 200px;  
  padding: 60px;  
}
```

# Formas de Adicionar CSS

Existem basicamente 4 formas de se adicionar CSS a um arquivo HTML. São estas: Inline, Tag Style, Tag Link e Import.

## Inline

```
<h1 style="font-size: 16px;">
  Olá, mundo!
</h1>
```

## Tag Link

```
<link rel="stylesheet" href="style.css">
```

## Tag Style

```
<head>
  <style>
    h1 {
      font-size: 32px;
    }
  </style>
</head>
<body>
  <h1>
    Olá, mundo!
  </h1>
</body>
```



# Formas de Adicionar CSS

Import – Em um arquivo .css, ou dentro da tag style, podemos inserir o @import no começo do arquivo desta forma:

```
@import url('https://fonts.googleapis.com/css2?family=Goblin+One&display=swap');
```

```
/* Ou */
```

```
@import 'https://fonts.googleapis.com/css2?family=Goblin+One&display=swap';
```

# A Cascata

A cascata tem a ver com a forma como o browser vai interpretar nossas declarações dentro do CSS. Se eu adicionar duas propriedades no mesmo elemento pelo CSS teremos um comportamento interessante:

```
h1 {  
  color: red;  
}  
  
h1 {  
  font-size: 10 px;  
}
```

Percebe-se que ele soma as duas alterações. Mas o que acontece se eu adicionar uma nova propriedade color?

# A Cascata

Percebemos que, desta forma, a cor que prevalece é a azul, dando sentido ao comportamento que comentamos anteriormente da cascata.

```
h1 {  
  color: red;  
}
```

```
h1 {  
  color: blue;  
}
```

# Hierarquia de Força


Existe uma "hierarquia de força" nas declarações de CSS, e estas são:  
Inline > Tag Style > Link.

Além disso, existe um cálculo matemático onde cada tipo de seletor e origem do estilo possuem valores a serem considerados:

- Universal selector, combinators e negation pseudo-class (:not())
- Element type selector e pseudo-elements (::before, ::after)
- Classes e attribute selectors ([type = "radio"])
- ID selector
- Inline

# A Regra Important

Não é considerada uma boa prática, devemos evitar o uso pois ela quebra o fluxo natural da cascata. Ela dá prioridade total sobre qualquer outra declaração de CSS.

```
h1 {  
  border: 1px solid  red !important;  
  margin: 200px;  
  padding: 60px;  
}
```

# At Rules (@)

```
@import url("http://local.com/style.css")

@media (min-width: 500px) {
  /* Regras aqui */
}

@font-face {
  /* Regras aqui */
}

@keyframes nomedaanimacao {
  /* Regras aqui */
}
```

Estão relacionadas ao comportamento do CSS. Começa com o sinal de @ seguido do identificador e valor. Alguns exemplos comuns são:

@import - Incluir um CSS externo

@media - Regras condicionais para diversos dispositivos

@font-face - Fontes externas

@keyframes - Animações

# Shorthand

Quando lidamos com algumas propriedades do CSS, podemos ter várias especificações para aquela propriedade, como por exemplo o **background**:

```
background-color: #000;  
background-image: url(images/bg.png);  
background-repeat: no-repeat;  
background-position: left top;
```

Podemos escrever todas essas propriedades utilizando o formato Shorthand desta forma:

```
background: #000 url(images/bg.png) no-repeat left top;
```

# Shorthand

Outro exemplo seriam as fontes:

```
font-style: italic;  
font-weight: bold;  
font-size: .8em;  
line-height: 1.2;  
font-family: Arial, sans-serif;  
  
font: italic bold .8em/1.2 Arial, sans-serif;
```

Alguns pontos devem ser considerados:

- Não serão consideradas as propriedades anteriores se houve especificações dos dois modelos;
- Valores não especificados assumirão o valor padrão;
- Geralmente a ordem descrita não importa mas se houver muitas propriedades com valores semelhantes, poderemos encontrar problemas.
- Saiba mais:  
[https://developer.mozilla.org/pt-BR/docs/Web/CSS/Shorthand\\_properties](https://developer.mozilla.org/pt-BR/docs/Web/CSS/Shorthand_properties)



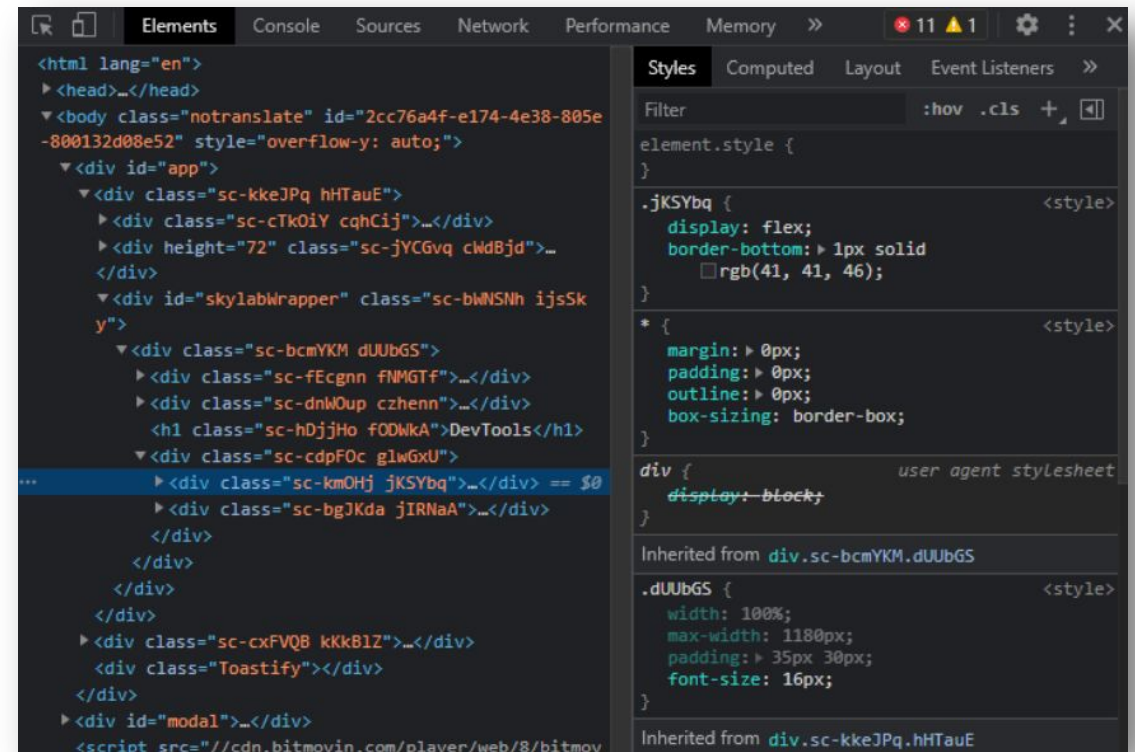
# Funções

Já utilizamos algumas funções por aqui, mas vamos enfatizá-las agora. Uma função é composta por um nome seguido de abre e fecha parêntesis. Além disso, podem receber argumentos dentro delas.

```
@import url("../")  
  
color: rgb(255, 0, 100);  
  
width: calc(100% - 10px);
```

# DevTools

Podemos acessar as propriedades de CSS de cada um dos elementos das páginas web, basta acessarmos as DevTools. Para isto, podemos clicar com o botão direito do mouse e selecionar "Inspecionar Elementos". Alguns browsers permitem acessar as DevTools através da tecla F12.



# Vendor Prefixes

Permite que os browsers adicionem novas features a fim de colocar em uso alguma novidade no CSS.

```
p {  
  -webkit-background-clip: text; /* Chrome, Safari, iOS e Android */  
  -moz-background-clip: text; /* Mozilla Firefox */  
  -ms-background-clip: text; /* Internet Explorer */  
  -o-background-clip: text; /* Opera */  
}
```

Maiores informações de compatibilidade podem ser encontradas [aqui](#).