



[rildo@uespi.br](mailto:rildo@uespi.br)

Tel: 86 98837-2010

# Programação Web Responsiva

## CSS



# Ajuda para esta aula

## Tabela de cores

- <https://site112.com/tabela-cores-html>

## Site com diversas imagens

- <https://pixabay.com/pt/>

## Site com imagens vetorizadas

- <https://br.freepik.com/>

# Enviando pasta e imagens para o github

## Pastas

Devemos sempre criar uma nova pasta com um arquivo qualquer clicando em create new file e dando o nome do arquivo com uma barra logo afrente do nome. Isso fará ser criado um diretório.

## Imagens

Enviar imagens é o mesmo método usado para envio de html. Entre no diretório desejado e escolha upload file.

# Estilizando com CSS

Quando escrevemos o HTML, marcamos o conteúdo da página com tags que melhor representam o significado daquele conteúdo. Aí quando abrimos a página no navegador é possível perceber que o navegador mostra as informações com estilos diferentes.

Um h1, por exemplo, fica em negrito numa fonte maior. Parágrafos de texto são espaçados entre si, e assim por diante. Isso quer dizer que o navegador tem um *estilo padrão* para as tags que usamos.

Antigamente, isso era feito no próprio HTML. Se quisesse um título em vermelho, era só fazer:

```
<h1><font color="red">Título do Site</font></h1>
```

Além da tag font, várias outras tags de estilo existiam.

# Estilizando com CSS

Em seu lugar, surgiu o **CSS**, que é uma outra linguagem, separada do HTML, com objetivo único de cuidar da estilização da página. A vantagem é que o CSS é bem mais robusto que o HTML para estilização, como veremos.

Mas, principalmente, escrever formatação visual misturado com conteúdo de texto no HTML se mostrou algo bem impraticável. O CSS resolve isso separando as coisas; regras de estilo não aparecem mais no HTML, apenas no CSS.

# Sintaxe e inclusão de CSS

A sintaxe do CSS tem estrutura simples: é uma declaração de propriedades e valores separados por um sinal de dois pontos ":", e cada propriedade é separada por um sinal de ponto e vírgula ";" da seguinte maneira:

**background-color: yellow;**

**color: blue;**

O elemento que receber essas propriedades será exibido com o texto na cor azul e com o fundo amarelo. Essas propriedades podem ser declaradas de três maneiras diferentes.

# Atributo style

A primeira delas é como um atributo style no próprio elemento:

`<p style="color: blue; background-color: yellow;">`

O conteúdo desta tag será exibido em azul com fundo amarelo no navegador! `</p>` Mas tínhamos acabado de discutir que uma das grandes vantagens do CSS era manter as regras de estilo fora do HTML.

Usando esse atributo style não parece que fizemos isso. Justamente por isso não se recomenda esse tipo de uso na prática, mas sim os que veremos a seguir.

# A tag <style>

A outra maneira de se utilizar o CSS é declarando suas propriedades dentro de uma tag <style>. Como estamos declarando as propriedades visuais de um elemento em outro lugar do nosso documento, precisamos indicar de alguma maneira a qual elemento nos referimos. Fazemos isso utilizando um **seletor CSS**. É basicamente uma forma de buscar certos elementos dentro da página que receberão as regras visuais que queremos.

Exemplo:

```
<style> h1 {color: blue; } </style>
```



# A tag <style> - exemplo

```
<!DOCTYPE html>
```

```
<html> <head>
```

```
<meta charset="utf-8"> <title>Site</title>
```

```
<style> p { background-color: yellow; color: blue; } </style>
```

```
</head>
```

```
<body>
```

```
<p> O conteúdo desta tag será exibido em azul com fundo amarelo!
```

```
</p>
```

```
</body> </html>
```

# Arquivo externo

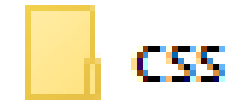
Outra maneira de declararmos os estilos do nosso documento é com um arquivo externo, geralmente com a extensão .css. Para que seja possível declarar nosso CSS em um arquivo à parte, precisamos indicar em nosso documento HTML uma ligação entre ele e a folha de estilo.

Além da melhor organização do projeto, a folha de estilo externa traz ainda as vantagens de manter nosso HTML mais limpo e do reaproveitamento de uma mesma folha de estilos para diversos documentos.

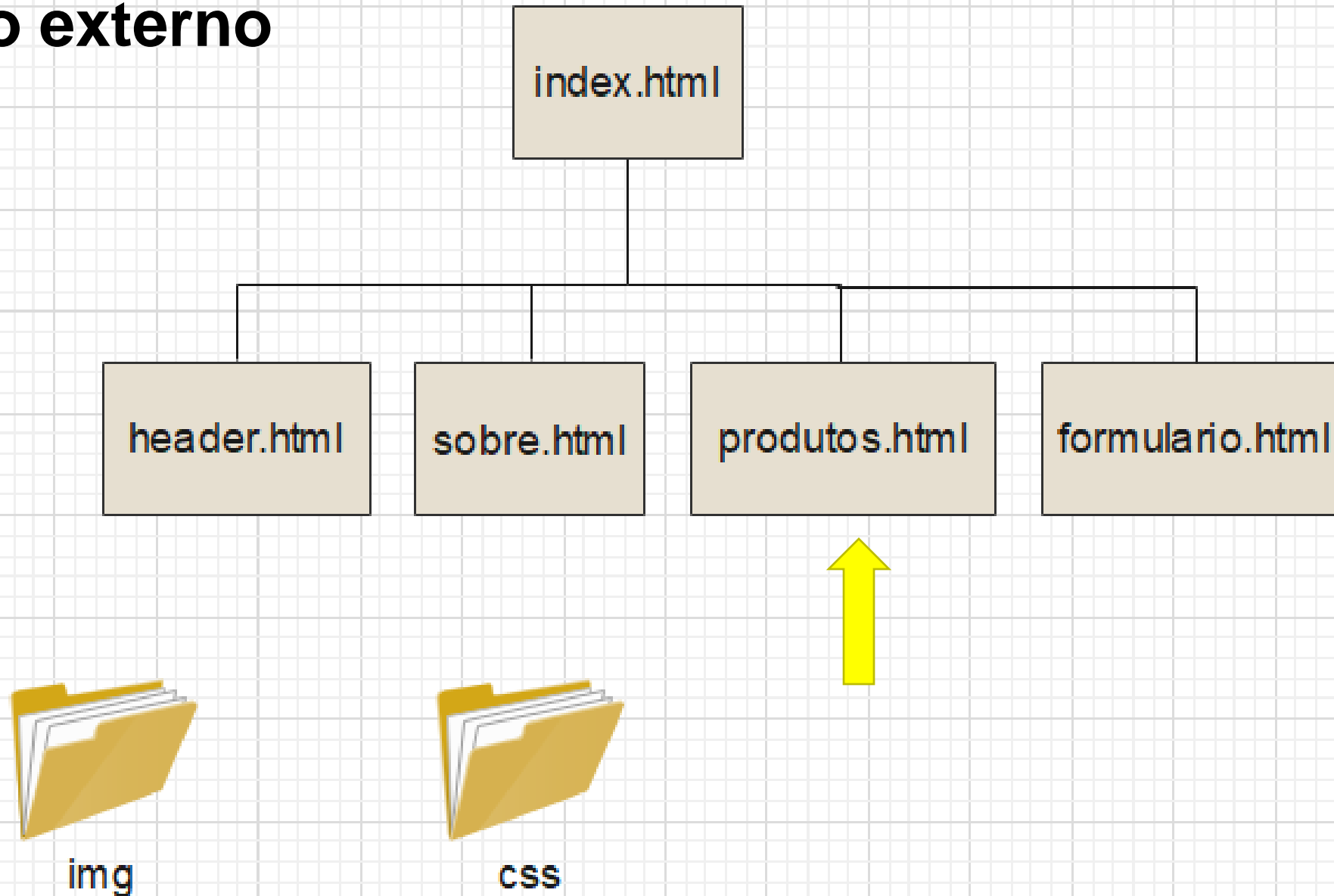
A indicação de uso de uma folha de estilos externa deve ser feita dentro da tag<head> do nosso documento HTML.

# Arquivo externo

Vamos criar nossa pasta de CSS



# Arquivo externo



# Arquivo externo

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8">  
<title>Site</title>
```

```
<link rel="stylesheet" href="css/estilos.css"> </head>
```

```
<body>
```

```
<p> O conteúdo desta tag será exibido em azul com fundo amarelo!  
</p>
```

```
</body>
```

```
</html>
```

E dentro do arquivo estilos.css colocamos apenas o conteúdo do CSS: **p { color: blue; background-color: yellow; }**

Vamos criar um arquivo estilo.css

# Arquivo externo

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Site</title>
<link rel="stylesheet" href="css/estilos.css">
</head>
<body>
<p> O conteúdo desta tag será exibido em azul com fundo amarelo! </p>
</body>
</html>
```

```
1 p { color: blue; background-color: yellow; }
2
3
4
```

# Propriedades tipográficas e fontes

Da mesma maneira que alteramos cores, podemos alterar o texto. Podemos definir fontes com o uso da propriedade font-family. A propriedade font-family pode receber seu valor com ou sem aspas. No primeiro caso, passaremos o nome do arquivo de fonte a ser utilizado, no último, passaremos a família da fonte. Por padrão, os navegadores mais conhecidos exibem texto em um tipo que conhecemos como "serif". As fontes mais conhecidas (e comumente utilizadas como padrão) são "Times" e "Times New Roman", dependendo do sistema operacional. Elas são chamadas de **fontes serifadas** pelos pequenos ornamentos em suas terminações.

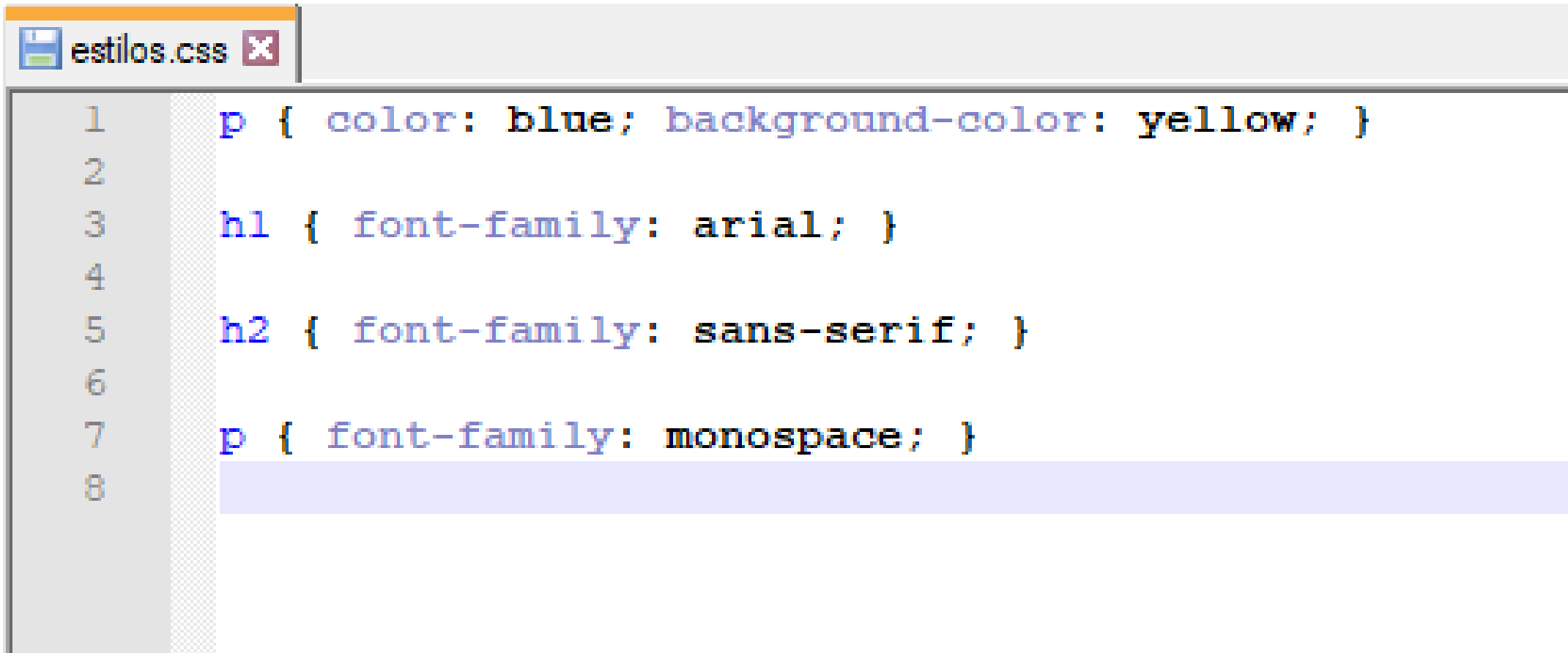
# Propriedades tipográficas e fontes

Podemos alterar a família de fontes que queremos utilizar em nosso documento para a família "sans-serif" (sem serifas), que contém, por exemplo, as fontes "Arial" e "Helvetica". Podemos também declarar que queremos utilizar uma família de fontes "monospace" como, por exemplo, a fonte "Courier".

- **h1 { font-family: arial; }**
- **h2 { font-family: sans-serif; }**
- **p { font-family: monospace; }**



# Propriedades tipográficas e fontes



```
1  p { color: blue; background-color: yellow; }
2
3  h1 { font-family: arial; }
4
5  h2 { font-family: sans-serif; }
6
7  p { font-family: monospace; }
8
```

O conteúdo desta tag será exibido em azul com fundo amarelo!

# Propriedades tipográficas e fontes

É possível, e muito comum, declararmos o nome de algumas fontes que gostaríamos de verificar se existem no computador, permitindo que tenhamos um controle melhor da forma como nosso texto será exibido. Normalmente, declaramos as fontes mais comuns, e existe um grupo de fontes que são consideradas "seguras" por serem bem populares.

Em nosso projeto, vemos que as fontes não têm ornamentos. Então vamos declarar essa propriedade para todo o documento por meio do seu elemento `body`:

```
body {  
font-family: "Arial", "Helvetica", sans-serif;  
}
```

# Propriedades tipográficas e fontes

Nesse caso, o navegador verificará se a fonte "Arial" está disponível e a utilizará para renderizar os textos de todos os elementos do nosso documento que, por cascata, herdarão essa propriedade do elemento body. Caso a fonte "Arial" não esteja disponível, o navegador verificará a disponibilidade da próxima fonte declarada, no nosso exemplo a "Helvetica". Caso o navegador não encontre também essa fonte, ele solicita qualquer fonte que pertença à família "sans-serif", declarada logo a seguir, e a utiliza para exibir o texto, não importa qual seja ela. Temos outras propriedades para manipular a fonte, como a propriedade font-style, que define o estilo da fonte que pode ser: normal (normal na vertical), italic (inclinada) e oblique (oblíqua).