Atividade 3 - Tutorial Guiado de Introdução ao CSS

Programação IV

24 de junho de 2025

Etapa 1 Criando sua primeira folha de estilos

Conceito

O CSS (*Cascading Style Sheets*) é utilizado para definir a aparência de páginas HTML. Ele permite controlar cor, fonte, espaçamento, bordas, posição e muitos outros aspectos da apresentação de uma página. Para que isso funcione, o CSS precisa estar associado ao documento HTML.

Formas de associação entre HTML e CSS

Existem três formas principais de aplicar CSS a uma página HTML:

1. **Estilo inline**: o estilo é escrito diretamente dentro do elemento HTML, utilizando o atributo `style`.

   Exemplo:

   ```
   <p style="color: red;">Este parágrafo está em vermelho.</p>
   ```

2. **Estilo interno**: o estilo é incluído dentro da própria página HTML, geralmente na seção `<head>`, usando a tag `<style>`.

   Exemplo:

   ```
   <head>
     <style>
       p {
         color: red;
       }
     </style>
   </head>
   <body>
     <p>Este parágrafo está em vermelho.</p>
   </body>
   ```

3. **Estilo externo (recomendado)**: o estilo é escrito em um arquivo separado com extensão `.css`, que é então vinculado ao HTML por meio da tag `<link>`.

   Exemplo (HTML):

```
<head>
  <link rel="stylesheet" href="estilo.css">
</head>
<body>
  <p>Este parágrafo será estilizado via CSS externo.</p>
</body>
```

   Exemplo (estilo.css):

```
p {
  color: red;
}
```

Sintaxe do CSS

Uma regra CSS é composta por um **seletor** seguido de um bloco de declarações. Cada **declaração** é formada por uma **propriedade** e um **valor**, separados por dois pontos, e terminada por ponto e vírgula.

Exemplo:

```
p {
  color: blue;
  font-size: 16px;
}
```

Neste exemplo:

- `p` é o **seletor**, indicando que a regra se aplica a todos os parágrafos (`<p>`).
- `color` e `font-size` são as **propriedades**.
- `blue` e `16px` são os **valores** atribuídos.

É possível aplicar múltiplas declarações dentro de um mesmo bloco, desde que cada uma esteja separada por ponto e vírgula.

Formato geral:

```
seletor {
  propriedade1: valor1;
  propriedade2: valor2;
}
```

Propriedades CSS Comuns

A tabela a seguir apresenta algumas das propriedades CSS mais utilizadas nesta etapa, com exemplos de valores válidos:

Cores de fundo

Propriedade	Exemplos de valores
`color`	`red`, `#333`, `rgb(0,0,0)`
`background-color`	`white`, `#f5f5f5`
`opacity`	`0.0` (transparente) a `1.0` (opaco)

Texto e fonte

Propriedade	Exemplos de valores
`font-family`	`Arial`, `"Times New Roman"`, `sans-serif`
`font-size`	`16px`, `1.2em`, `120%`
`font-weight`	`normal`, `bold`, `lighter`
`text-align`	`left`, `center`, `justify`
`text-decoration`	`none`, `underline`, `line-through`

Tarefa prática

Crie dois arquivos na mesma pasta:

- Um arquivo HTML que contenha:
  - Um título descritivo da página (exibido na aba do navegador);
  - Um cabeçalho de primeiro nível com uma saudação ou título;
  - Um parágrafo de introdução.
- Um arquivo CSS (por exemplo, `estilo.css`) que defina:
  - Uma cor de fundo clara para a página;
  - Uma cor distinta para o cabeçalho;
  - Um estilo de fonte para o parágrafo.

Associe a folha de estilos externa ao HTML utilizando a forma mais adequada para projetos profissionais (arquivo externo com a tag `<link>`).

Abra o HTML em um navegador e observe as mudanças visuais produzidas pelo CSS.

Verificação de Aprendizagem

1. Qual das formas abaixo representa a maneira mais recomendada de aplicar CSS em projetos profissionais?

(A) Estilo inline

(B) Estilo interno

(C) Estilo externo

(D) Nenhuma das anteriores

2. Qual das opções abaixo representa corretamente uma declaração CSS?

(A) `color = blue;`

(B) `p(color: blue)`

(C) `color: blue;`

(D) `p.color: blue`

3. A propriedade `font-family` serve para:

(A) Mudar a cor da fonte

(B) Definir o tamanho da fonte

(C) Escolher o estilo de fonte do texto

(D) Justificar o texto

4. A propriedade `background-color` pode receber valores como:

(A) Apenas nomes de cores

(B) Apenas valores em hexadecimal

(C) Apenas códigos RGB

(D) Todas as anteriores

Etapa 2 Seletores e Regras CSS

Conceito

CSS funciona por meio de regras que associam um **seletor** (o elemento HTML a ser estilizado) a um conjunto de **declarações** (estilos aplicados ao elemento). Uma declaração é composta por uma **propriedade** e um **valor**.

O seletor define *quem* será afetado, e as declarações definem *como* ele será exibido.

Exemplos de seletores comuns:

- **De tipo:** afeta todas as tags de um determinado tipo (por exemplo, todos os parágrafos `<p>`).
- **De classe:** afeta todos os elementos com o atributo `class` correspondente. Inicia com ponto (ex: `.destaque` - afetaria tags com `class='destaque'`).
- **De ID:** afeta um único elemento com um identificador único. Inicia com cerquilha (ex: `#principal` - afetaria tags com `id='principal'`).

Prioridade de aplicação das regras CSS

Quando múltiplas regras CSS se aplicam ao mesmo elemento, o navegador segue critérios para decidir qual estilo prevalecerá. Esses critérios seguem uma ordem de prioridade:

1. **Estilo inline** (escrito diretamente na tag HTML com `style="..."`) tem a **maior prioridade**.
2. **Seletor de ID** tem prioridade maior que seletores de classe e de tipo.
3. **Seletor de classe** tem prioridade maior que seletores de tipo.
4. **Seletor de tipo** (como `p`, `h1`) tem menor prioridade entre os estilos declarados em arquivos CSS externos ou internos.
5. **Última regra válida**: se duas regras tiverem a mesma especificidade, prevalece a que estiver mais abaixo no código CSS.

Exemplo de ordem de prioridade (da maior para a menor):

```
style="..." > #id > .classe > tipo (ex: p)
```

Além disso, algumas propriedades CSS (como `color` ou `font-family`) podem ser herdadas de elementos "pais" no HTML, enquanto outras (como `margin` ou `border`) não são herdadas automaticamente.

—

Dica: Evite usar `style="..."` diretamente no HTML. Prefira usar seletores de classe ou ID em folhas de estilo externas, para manter seu código organizado e reutilizável.

Tarefa prática

Adicione novos elementos ao seu arquivo HTML com diferentes tipos de seletores. Por exemplo:

- Um parágrafo com uma classe chamada `destaque`.
- Um parágrafo com um `id` específico, como `id="aviso"`.
- Um título do tipo `<h2>` sem classe nem id.

No arquivo CSS, defina estilos diferentes para:

- Todos os elementos `<h2>` (seletor de tipo).
- A classe `destaque`, com uma cor e negrito.
- O ID `aviso`, com uma cor diferente e estilo itálico.

Abra o arquivo HTML no navegador e observe as diferenças de estilo aplicadas a cada elemento, conforme o seletor utilizado.

Verificação de Aprendizagem

1. Qual das alternativas representa um seletor de classe válido?

(A) `p`
(B) `#destaque`
(C) `.destaque`
(D) `destaque`

2. Assinale a alternativa que indica corretamente a ordem de prioridade entre seletores, do mais forte para o mais fraco:

(A) $Classe > ID > Tipo$
(B) $Tipo > Classe > ID$
(C) $ID > Classe > Tipo$
(D) $Inline > Tipo > ID$

3. Suponha que um elemento HTML tenha um estilo definido por três regras conflitantes:

- Uma regra de tipo: `p { color: blue; }`
- Uma regra de classe: `.mensagem { color: green; }`
- Uma regra de ID: `#alerta { color: red; }`

Se o elemento tiver `class="mensagem"` e `id="alerta"`, qual cor será aplicada ao texto? Justifique com base na especificidade dos seletores.

4. Um aluno escreveu o seguinte código CSS:

```
.texto {
  font-size: 18px;
  color: blue;
}

p {
  color: green;
}
```

Sabendo que um elemento do tipo `<p>` também tem `class="texto"`, qual será a cor final do texto?

(A) Azul

(B) Verde

(C) Depende da ordem

(D) Não será aplicada nenhuma cor

Etapa 3 Modelo de Caixa (Box Model)

Conceito

No CSS, todo elemento HTML é tratado como uma **caixa retangular**, mesmo que visualmente pareça diferente. Esse conceito é conhecido como **Modelo de Caixa (Box Model)** e é fundamental para organizar o layout de uma página.

Cada caixa é composta por quatro partes, de dentro para fora:

1. **Conteúdo (content):** o espaço onde ficam o texto ou os elementos internos.
2. **Preenchimento (padding):** espaço entre o conteúdo e a borda da caixa. O `padding` afasta o conteúdo da borda interna.
3. **Borda (border):** a linha que envolve o conteúdo e o preenchimento. Pode ter cor, largura e estilo.
4. **Margem (margin):** espaço externo que separa a caixa de outros elementos ao redor. O `margin` não afeta o conteúdo interno.

Outras propriedades associadas

- `width` e `height`: definem a largura e altura da área de conteúdo.
- `margin: auto`: pode ser usada para centralizar um bloco horizontalmente (quando há largura definida).
- `padding`, `border` e `margin` influenciam o espaço total ocupado pela caixa.
- A **largura total visível da caixa** será: `width` + `padding` + `border`. A margem não entra nesse cálculo.

Exemplo de cálculo de largura total:
Se definirmos:

```
width: 300px;
padding: 20px;
border: 5px solid black;
```

A largura total visível da caixa será:

```
300 (conteúdo) + 40 (20 esquerda + 20 direita) + 10 (5 esquerda + 5 direita) = 350px
```

Importante: por padrão, o navegador soma essas dimensões. Essa abordagem pode ser alterada com a propriedade `box-sizing`, que será discutida mais adiante.

Tarefa prática

- Crie uma `<div>` com um parágrafo dentro dela.
- Defina uma largura fixa (`width`) para a `div`.
- Adicione `padding` para afastar o texto da borda interna.
- Adicione uma `border` com cor e espessura visíveis.
- Aplique `margin` para afastar essa caixa de outros elementos da página.
- Teste o valor `margin: 0 auto` e observe a centralização do elemento.
- Observe no navegador como cada valor afeta visualmente a caixa e sua posição.

Verificação de Aprendizagem

1. Suponha que você aplicou a seguinte regra a uma `div`:

```
div.bloco {
  padding: 30px;
}
```

O que será afetado por essa regra?

(A) A distância entre essa `div` e os elementos ao redor.
(B) O espaço entre o conteúdo da `div` e sua borda.
(C) A espessura da borda.
(D) A largura total da página.

2. Um aluno usou a seguinte declaração:

```
margin: 0 auto;
```

Qual o efeito visual dessa regra, quando aplicada a um elemento com largura definida?

(A) O elemento ficará alinhado à esquerda.
(B) A margem superior será removida.
(C) O elemento será centralizado horizontalmente.
(D) O conteúdo interno terá espaçamento uniforme.

3. Observe a regra a seguir:

```
.caixa {
  width: 300px;
  padding: 20px;
  border: 5px solid black;
}
```

Qual será a largura total visível (externa) da caixa, desconsiderando margens?

(A) 300px

(B) 340px

(C) 350px

(D) 370px

4. Considere o seguinte código:

```
.caixa {
  border: 2px solid black;
  padding: 20px;
}
```

O que aconteceria se substituíssemos `padding` por `margin` com o mesmo valor?

(A) O conteúdo interno ficaria mais afastado da borda.

(B) O distanciamento entre elementos externos aumentaria.

(C) A borda ficaria mais espessa.

(D) Nada mudaria visivelmente.