

A faint, light gray cartoon illustration of a person with glasses, smiling and holding a large book. The person is wearing a simple shirt. The background is white with some faint geometric shapes like a diamond and a circle.

Linguagem de marcação e estilos - CSS

Você vai compreender o que são folhas de estilo (CSS) e como podem ser utilizadas para cuidar da apresentação, layout e estilo de páginas HTML.

Prof. Alexandre Paixão

1. Itens iniciais

Propósito

Para aplicação dos exemplos, será necessário um editor de texto com suporte à marcação HTML. No sistema operacional Windows, é indicado o Notepad++; no Linux, o Nano Editor.

Objetivos


- Identificar os fundamentos da CSS.
- Reconhecer os recursos de cores, texto, fontes e web fontes da CSS3.
- Identificar os conceitos de box model, pseudoclasses, pseudoelementos e posicionamento.
- Reconhecer frameworks CSS.

Introdução

A CSS, ou folhas de estilo em cascata (cascading style sheets), é uma linguagem de estilo que fornece total controle sobre a apresentação de um documento escrito em HTML.

No ambiente web, a HTML é a linguagem responsável pela estrutura do conteúdo de uma página. Embora também seja capaz de organizar o conteúdo visualmente, é função da CSS cuidar desse aspecto e de tudo relacionado ao estilo e layout da página.

Com a CSS, é possível, por exemplo, alterar a forma e o posicionamento dos elementos, as cores, tipos e tamanhos de fontes, e muito mais.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

1. Fundamentos da CSS

Como a CSS funciona?

A CSS é uma linguagem de estilo que fornece total controle sobre a apresentação de um documento escrito em HTML. Com ela, é possível alterar, entre outras coisas, a forma e o posicionamento dos elementos, as cores, os tipos e tamanhos de fontes.

Neste vídeo, você vai entender como funcionam as folhas de estilos CSS, incluindo sintaxe, seletores e propriedades. Além disso, você vai compreender a relação entre CSS e HTML.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Sintaxe da CSS

A CSS permite a aplicação seletiva de estilos a elementos em uma página HTML. Ou seja, um ou mais estilos podem ser aplicados em um documento inteiro ou mesmo em apenas parte dele. Além disso, um mesmo tipo de elemento pode ter, ao longo do documento, diferentes estilos.

Para aplicar um estilo CSS a um elemento específico, é necessário **identificá-lo** e apontar qual de suas **propriedades** queremos alterar e qual **valor** queremos atribuir a ele. Essas três informações definem a sintaxe da CSS.

Veja o código a seguir. Consegue identificar o seletor, a propriedade e o valor?

```
p{
  background-color: blue;
}
```

Sintaxe da CSS

Vamos ver agora cada uma das informações necessárias para aplicarmos um estilo utilizando CSS apontadas na imagem:

O seletor

Nesse caso, a tag HTML `<p>`.

```
Seletor
p{
  background-color: blue;
}
```

Ao menos uma propriedade

Nesse caso, a cor de fundo (**background-color**).

```
p{
  background-color: blue;
}
Propriedade
```

Ao menos um valor para a propriedade

Nesse caso, a cor azul (**blue**).

```
p{
  background-color: blue;
}
Valor
```

Essa declaração de estilo faria com que todas as tags `<p>` do documento apresentassem a cor azul ao fundo.

Além disso, há outros aspectos importantes com relação à **sintaxe**:

- A propriedade e seu valor devem ser separados por dois pontos :
- Uma declaração deve ser separada da declaração subsequente com utilização do ponto e vírgula ;
- O conjunto de estilos aplicados a um seletor é envolvido por chaves {...}
- É possível inserir várias declarações (propriedade + valor) em conjunto

Veja um exemplo de duas propriedades atribuídas à tag `<p>` e o resultado dessas declarações no navegador.

```
p{
  background-color: blue;
  color: white;
}
```

Texto do parágrafo estilizado com CSS


Demonstração da aplicação de estilos.

Seletores

Nos exemplos anteriores, um estilo foi declarado em uma tag HTML `<p>`. Nós nos referimos a essa tag como sendo o **seletor** ao qual o estilo foi aplicado. Existem muitos outros seletores disponíveis além daqueles correspondentes às tags HTML.

Seletores class e id

O **seletor de classe** é definido a partir da declaração do atributo **class** em um elemento HTML. Já o **seletor de identificação** é definido com o atributo **id**.



Atenção

Não existe uma regra a ser seguida em termos de nomenclatura para a definição do nome da classe ou do identificador. Procure utilizar nomes que tenham relação com a função do elemento na página e que, de fato, ajudem a identificá-lo ou classificá-lo.

No fragmento de código seguinte, veja um exemplo de sintaxe correspondente à declaração desses atributos na HTML.

```
...
<h1 class="texto_vermelho">Título Principal</h1>
<p id="texto_apresentacao">
  Texto do parágrafo com atributo de identificação.
</p>
<h2>Primeiro Subtítulo</h2>
<p class="texto_descricao texto_vermelho">
  Texto do parágrafo com atributo de classe.
</p>
<h2>Segundo Subtítulo</h2>
<p class="texto_descricao">
  Texto do parágrafo com atributo de classe.
</p>
...
```

Sintaxe de declaração de seletores de classe e identificação na HTML

Na imagem, observamos que uma mesma classe, a “texto vermelho”, foi atribuída à tag `<h1>` e a uma das tags `<p>`. Com isso, vemos que uma classe pode ser atribuída a mais de um elemento. Em seguida, note a sintaxe para atribuição de múltiplas classes a um elemento na segunda tag `<p>` à qual foram atribuídas as classes “texto_descricao” e “texto_vermelho”.

Observe agora o código CSS:

```
<style type="text/css">
#texto_apresentacao{
|   font-size: 16px;
}

.texto_descricao{
|   font-size: 12px;
}

.texto_vermelho{
|   color: ■ red;
}
</style>
```

Primeira forma de sintaxe CSS

```
<style type="text/css">
p#texto_apresentacao{
|   font-size: 16px;
}

p.texto_descricao{
|   font-size: 12px;
}

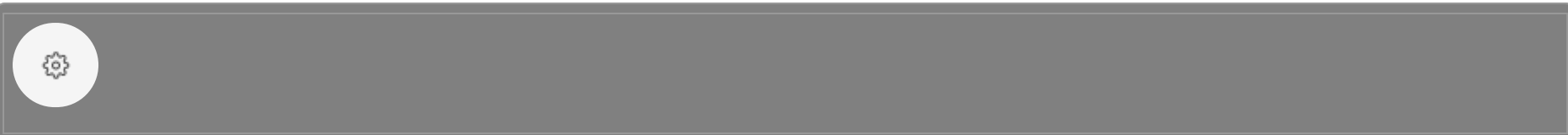
.texto_vermelho{
|   color: ■ red;
}
</style>
```

Segunda forma de sintaxe CSS

Veja que o seletor de id é representado por uma “#” e o de class é representado por um “.”, sendo seguidos de seus nomes. Além disso, foram apresentadas duas formas de sintaxe que produzirão o mesmo efeito. A diferença entre ambas é que, na segunda, o nome da tag à qual a identificação ou classe foi atribuída precede o respectivo sinal.

Restrições e boas práticas na utilização do identificador

Embora não exista um padrão ou preferência para utilização do seletor id ou class, é importante frisar que um id deve ser aplicado a **apenas um elemento**, enquanto a class pode ser aplicada a **um ou vários elementos**.



Dica

O navegador não verifica se um mesmo id foi utilizado em diferentes elementos. Porém, como esse seletor também é bastante usado pelo Javascript, podem ocorrer problemas de estilização e comportamento. Por isso, defina identificadores únicos.

Seletores de atributo

Utilizam nomes de atributos dentro de colchetes, sendo possível combiná-los com valores. Observe alguns dos seletores de atributo disponíveis:

[checked]

Seleciona todos os elementos que

[type='text']

Seleciona todos os elementos do tipo

Os seletores de atributo são flexíveis, permitindo inúmeras combinações. Por exemplo, é possível usá-los para selecionar todas as imagens com determinada extensão, selecionar todos os elementos com o atributo title contendo determinado valor etc.

Seletores baseados em relacionamento

É possível declarar estilos utilizando a relação entre os elementos. Confira os principais seletores baseados em relacionamento:

H1 P

Qualquer elemento P que seja

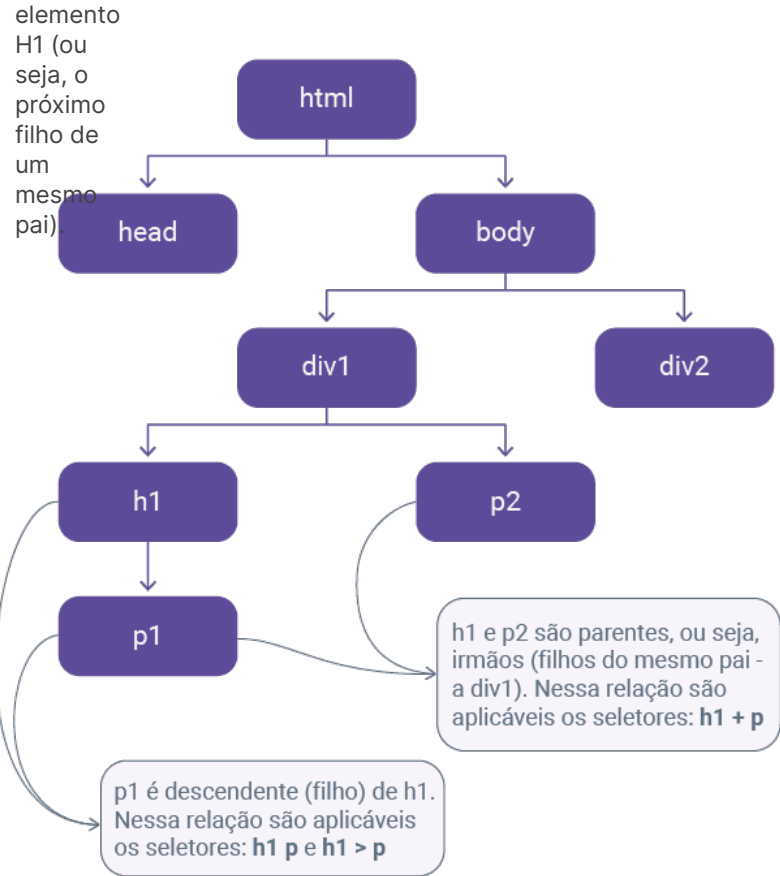
H1 > P

Qualquer elemento P que seja filho de um elemento H1.

H1 + P

Qualquer elemento P que seja o

Para uma melhor compreensão quanto à descendência e ao parentesco mencionados, veja uma representação simbólica da árvore DOM (representação estruturada do documento HTML em formato de árvore) contendo elementos representando tags HTML. Veja ainda a explicação de cada relação e aplicação dos seletores baseados em relacionamento.



Relação entre elementos em uma página HTML

Propriedades CSS

Existem inúmeras propriedades CSS, desde as definidas pela sua especificação, ditas **propriedades padrão**, até as **proprietárias**, que funcionam apenas em alguns navegadores. A fim de garantir uma maior compatibilidade, assim como otimizar o desenvolvimento, devemos sempre dar preferência às propriedades padrão. Estas são algumas das propriedades mais comuns da CSS:

background

Estiliza o fundo de elementos. Para tal, há uma série de propriedades, além do atalho “background”, como “background-color”, “background-image” etc.

border

Controla as bordas de um elemento, sendo possível definir suas cores, espessuras, entre outras propriedades.

top, bottom, right e left

Controlam o posicionamento, relativo ou absoluto, dos elementos em relação a outros elementos.

color

Estiliza a cor do conteúdo textual de um elemento.

font-family, font-size, font-weight etc.

Série de propriedades usada para estilizar o conteúdo textual de um elemento no que diz respeito à fonte, como a família de fontes, seu tamanho, peso (mais clara ou mais escura - negrito) etc.

height

Define a altura de um elemento.

list-style, list-style-image etc.

Propriedades usadas para estilizar as listas HTML.

margin

Controla a distância em função da margem de um elemento para outro.

padding

Controla a distância entre as bordas e o conteúdo de um elemento.

position
Define como um elemento deve ser posicionado na página.
text-...
Muitas propriedades controlam o comportamento do conteúdo textual de um elemento, como alinhamento (justificado, centralizado etc.), aparência (sublinhado etc.) etc.
width
Define a largura de um elemento.
z-index
Define a profundidade de um elemento – usado, por exemplo, para sobreposição de elementos.

CSS e HTML

Há três formas usuais de aplicar estilos em um documento HTML usando CSS: **Inline**, **Interna** e **Externa**. Além dessas, a HTML5 permite a aplicação **em escopo**.

CSS inline

Essa forma envolve a declaração do estilo CSS diretamente na tag, no código HTML. A declaração inline faz uso do atributo style procedido por declarações separadas por ponto e vírgula “;”. Esse atributo pode ser usado em qualquer tag HTML.

```
<p style="background-color: blue;">Texto parágrafo</p>
```

Aplicação de estilo inline.

CSS interna

Também chamada de CSS incorporada, é declarada na seção <head> do documento HTML.

```
<html>
...<head>
...  <style type="text/css">
...    p{
...      background-color: blue;
...    }
...  </style>
...
...</head>
...
</html>
```

Aplicação de estilo CSS interna

CSS externa

Nesse caso, os estilos são declarados em um arquivo externo, com extensão “.css” e vinculados ao documento HTML por meio da tag **<link>** ou da diretiva @import dentro da tag <head>.


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

<html>
... <head>
... <style>
... @import url("estilos.css");
... </style>
...
... </head>
...
</html>
Aplicação de estilo externa
```

CSS em escopo

Criada a partir da HTML5, por meio dessa forma de aplicação de estilo, é possível aplicar estilos no âmbito de escopo, ou seja, específicos para as seções da página em que foram declarados, incluindo os seus elementos filhos. No código a seguir, a tag <p> receberá os estilos definidos, sendo a mesma regra válida para outros estilos e elementos que, porventura, venham a fazer parte da <div>.

```
...
<div>

... <style type="text/css">
... /* Estes estilos serão aplicados apenas dentro da Div */
... p{
... background-color: blue;
... }
... </style>
... <p>Texto do parágrafo</p>

</div>
...
Aplicação de estilo a nível de escopo
```

Efeito cascata

Quando trabalhamos com CSS, é comum nos depararmos com a declaração conflitante de estilos, ou seja, diferentes definições de estilo para um mesmo elemento. Nessas situações, entra em ação o efeito cascata. Para entendermos a definição desse efeito, é preciso abordarmos dois conceitos: herança e especificidade.

Herança

A CSS permite que a aplicação de propriedades a elementos pais seja herdada pelos seus elementos filhos. Tomemos como exemplo o código a seguir.

```
...
div{
... color: blue;
}

<div>
... Texto solto na Div
... <p>Texto do parágrafo que é "filho" da Div</p>
</div>

...
Exemplo de herança em CSS
```

O resultado do fragmento de código mostrará tanto o texto solto quanto o texto dentro da tag <p> com a cor azul. Isso significa que a tag <p> herdou o estilo definido para o seu pai, a tag <div>.

A capacidade de herdar estilos caracteriza o que chamamos de efeito cascata.

Nem todas as propriedades CSS podem ser herdadas pelos filhos! Um exemplo são as propriedades relacionadas à formatação das boxes (que veremos mais adiante), como largura, altura, entre outras.

Especificidade

Para entender o que é a especificidade no âmbito das folhas de estilo, vamos recorrer a mais um exemplo.

```
...
div{
... color: blue;
}

<div>
... Texto solto na Div
... <p>Texto do parágrafo que é "filho" da Div</p>
</div>

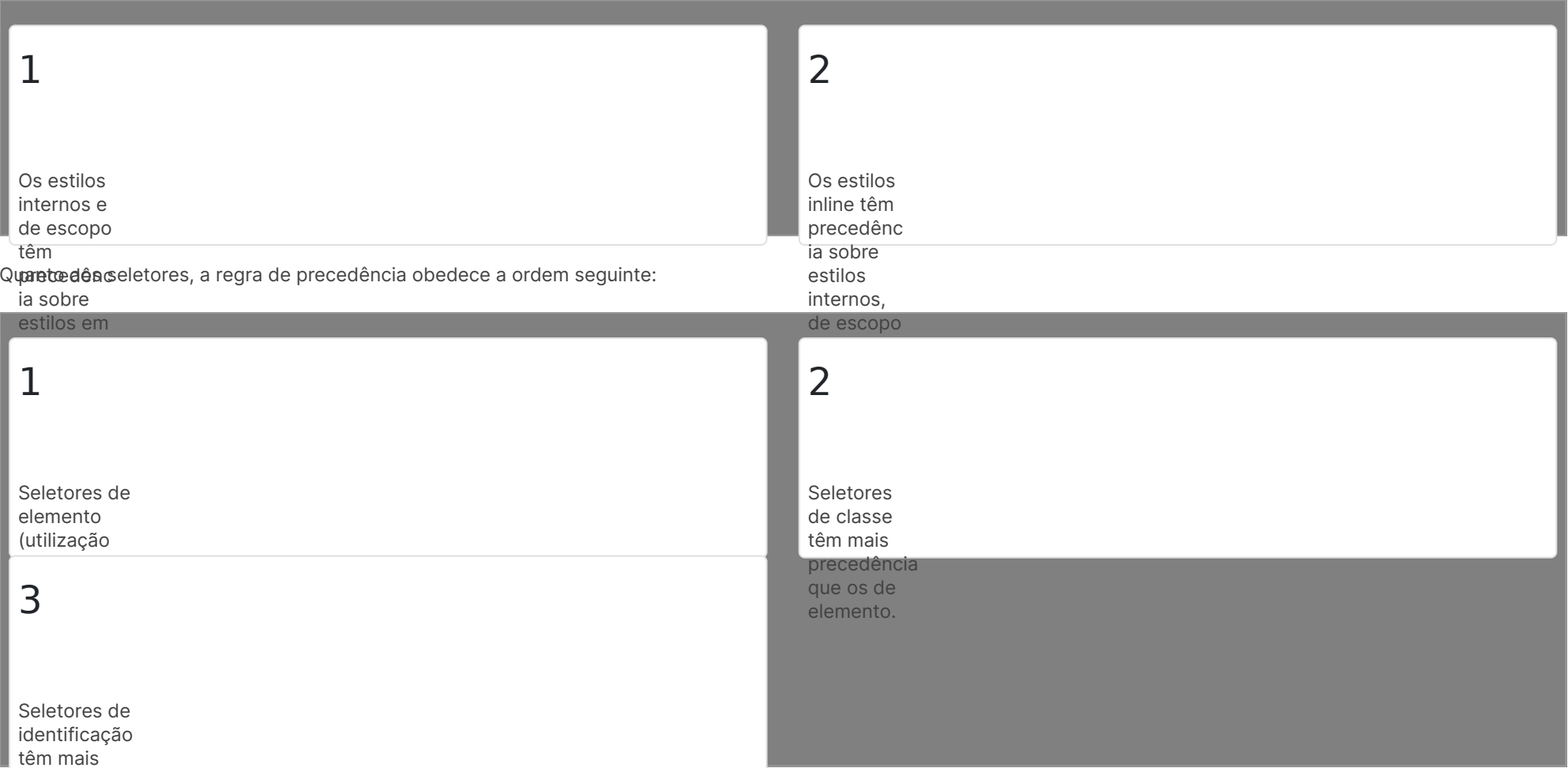
...
Exemplo de especificidade em CSS.
```

Perceba que o fragmento da imagem “Exemplo de herança em CSS” foi adaptado. Antes, era aplicado o conceito de herança, assim, o texto dentro da tag filha assumia o estilo definido para o seu pai. Agora, há um **estilo específico** definido para todas as tags **<p>** que sejam filhas de tags **<div>**. Com isso, ao visualizarmos o resultado no navegador, teremos o texto solto na cor azul e o texto dentro da tag **<p>** na cor vermelha.

A CSS é bastante flexível e nos permite definir diferentes níveis de especificidade. Entretanto, é importante termos cuidado com a sobreposição de estilos, ou seja, diferentes estilos definidos para um mesmo elemento em diferentes partes de nosso código CSS. A regra nesse caso é: prevalecerá o estilo mais específico. No exemplo anterior, a primeira declaração (para a tag **div**) é generalizada; a segunda (**div p**), específica.

Regras de precedência

Em relação às formas de inclusão da CSS, a regra de precedência obedece a seguinte ordem:



Veja este novo exemplo:
...
p {
 color: blue !important;
}

div p {
 color: red;
}
...
<div>
 Texto solto na Div
 <p>Texto do parágrafo que é "filho" da Div</p>
</div>

```
...  
p {  
  color: blue !important;  
}  
  
div p {  
  color: red;  
}  
...  
<div>  
  Texto solto na Div  
  <p>Texto do parágrafo que é "filho" da Div</p>  
</div>
```

Exemplo de aplicação do valor !important

Seguindo as regras de especificidade, sua aplicação resultaria na apresentação do texto dentro da tag **<p>** na cor vermelha, pois sua declaração de estilo é mais específica que a utilizada para a tag **<p>** (texto azul), que é generalizada. Entretanto, a utilização do valor **!important**, que se enquadra no que chamados de **CSS hack**, na declaração mais generalizada, faz com que esse estilo se sobreponha ao específico. Logo, o código anterior resulta na apresentação do texto dentro da tag **<p>** na cor azul.

CSS hack

Técnica de codificação usada para alterar o comportamento natural da CSS e ocultar ou mostrar determinada declaração de acordo com o navegador, sua versão ou recursos disponíveis.

Abreviaturas ou atalhos em CSS

As folhas de estilo permitem a aplicação de algumas propriedades utilizando abreviaturas ou atalhos. O exemplo a seguir mostra as duas formas de realizar uma mesma declaração:

<pre>p{ ;margin-top: 10px; margin-bottom: 8px; margin-right: 6px; margin-left: 4px; }</pre>	<pre>p{ margin: 10px 6px 8px 4px; }</pre>
<pre>p{ padding-top: 10px; padding-bottom: 8px; padding-right: 6px; padding-left: 4px; }</pre>	<pre>p{ padding: 10px 6px 8px 4px; }</pre>
<pre>p{ border-width: 2px; border-style: solid; border-color: #cccccc; }</pre>	<pre>p{ border: 2px solid #cccccc; }</pre>
<pre>p{ background-color: #000000; background-image: url(imagem.jpg); background-repeat: no-repeat; background-position: top left; }</pre>	<pre>p{ background: #000000 url(imagem.jpg) no-repeat top left; }</pre>
<pre>p{ font-size: 1em; line-height: 1.5em; font-weight: bold; font-style: italic; font-family: verdana; }</pre>	<pre>p{ 1em/1.5em bold italic verdana; }</pre>
<pre>ul{ list-style: #000000; list-style-type: disc; list-style-position: outside; list-style-image: url(imagem.jpg); }</pre>	<pre>ul{ list-style: disc outside url(imagem.jpg); }</pre>

Duas formas de realizar a mesma declaração
Alexandre Paixão

CSS3

A CSS, atualmente, está em sua terceira versão. Dentre as diversas novidades dessa versão, destacam-se:

- Melhorias nos seletores, com novas possibilidades de seleção: primeiro e/ou último elemento, elementos pares ou ímpares etc.
- Efeito gradiente e de sombra em textos e elementos.
- Bordas arredondadas.
- Manipulação de opacidade.
- Controle de rotação e perspectiva.
- Animações.

Atividade

A respeito da integração HTML e CSS, assinale a afirmativa correta:

A

Tanto a HTML quanto a CSS são renderizadas pelo navegador que, interpretando as tags de marcação e os estilos que lhes são aplicados, as exibe em tempo de execução/requisição pelo usuário.

B

Todo o código CSS é compilado pelo servidor web que o transforma em código HTML nativo a fim de que possa ser exibido no navegador.

C

A CSS inline, incorporada e de escopo são renderizadas diretamente pelo navegador, juntamente com a HTML. Já a CSS externa, por não estar dentro do arquivo HTML, precisa ser compilada pelo servidor web antes de ser renderizada.

D

Apenas a partir da HTML5, com a possibilidade de declaração de estilos em escopo, os navegadores passaram a dar suporte à renderização da CSS e do HTML sem necessidade de compilação.

E

A HTML provê, basicamente, duas formas para aplicação de estilos: CSS inline e CSS interna. Somente a partir da HTML5 mais duas formas foram disponibilizadas: CSS externa e CSS de escopo.

A alternativa A está correta.

Tanto a HTML quanto a CSS são linguagens interpretadas diretamente pelo browser e que não precisam ser compiladas – exceto a CSS quando se utiliza pré-processadores.

Estilizando páginas HTML com CSS

Tendo visto os fundamentos da CSS, chegou a hora de colocar o conhecimento adquirido em prática. Nesse contexto, aplicaremos as folhas de estilo em uma página HTML.

Confira neste vídeo a estilização de páginas HTML usando a linguagem de marcação e estilos (CSS).



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Roteiro de prática

Você aplicará alguns dos conceitos vistos sobre os fundamentos da CSS – mais precisamente, aplicará a estilização de texto utilizando o seletor de ID. Para isso, em termos de ferramentas, precisará do seguinte: um editor de textos – pode ser o próprio bloco de notas do Windows ou o Nano Editor do Linux, ou então algo um pouquinho mais avançado, como o software (livre/gratuito) Notepad++ e também de um navegador – Google Chrome, Firefox, MS Edge etc.

Vamos ao roteiro!

- No editor, crie a estrutura base de uma página HTML. Dentro da tag <body>, insira uma tag <p> e algumas linhas de texto dentro dela.
- Atribua um ID para essa tag <p>.

- Insira, de forma inline, alguns estilos para modificar a aparência do texto: modifique o tamanho da fonte e também a sua cor, por exemplo.
- Salve a página no editor e a abra no navegador para ver o resultado.

Como resolução, teremos o código a seguir.

plain-text

```

    Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    In euismod nibh sit amet justo cursus, eget posuere arcu pharetra.
    Donec vestibulum, lectus non consectetur condimentum, ex purus
    luctus lectus, at lacinia augue sem euismod diam.
    Aliquam aliquam accumsan varius. Fusce facilisis sollicitudin tortor.
    Curabitur aliquet bibendum ligula, vitae porta nunc ornare sed.
    Aliquam dapibus scelerisque turpis, eget hendrerit turpis elementum in.
    Aliquam accumsan sit amet nisi euismod imperdiet.
    Integer non pellentesque ante, nec gravida leo.
```

Faça você mesmo!

As folhas de estilo (CSS) permitem estilizar os elementos de uma página HTML. Focando, inicialmente, em estilizações mais simples, vamos abordar algumas outras formas de modificarmos textos de uma página. Considere as opções e assinale a alternativa correta, que permita destacar em negrito – seletor e respectiva propriedade –, o texto dentro de uma tag <div> com o seletor do tipo classe, com o nome de “texto-apresentacao”.

☐

A

#texto-apresentacao{ Strong; }

☐

B

.texto-apresentacao{ font-weight: bold; }

☐

C

texto-apresentacao

☐

D

#texto-apresentacao{ text-weight: bold; }

☐

E

.texto-apresentacao{ text-weight: strong; }

☒

✓

A alternativa B está correta.

Embora seja possível destacar um texto como negrito utilizando apenas HTML e as tags ou , a CSS possui a propriedade “font-weight” para tal finalidade e que deve ser sempre utilizada, salvo situações de acessibilidade (permitir a leitores de tela lerem, com ênfase, o texto englobado pela tag). Tal propriedade possui como valores aplicáveis diferentes níveis (variando entre 400 e 700), como a opção “bold”, da alternativa B.

2. Tecnologia CSS3

Recursos da CSS3

A CSS tem evoluído ao longo dos anos, como as tecnologias de forma geral, passando por diferentes versões/especificações. Sua versão mais atual, a CSS3, trouxe como novidades recursos interessantes voltados para estilização de cores, textos e fontes, por exemplo. Ao longo deste módulo, veremos algumas dessas novidades.

Conheça neste vídeo o CSS3 e os seus recursos de cores, textos, fontes e web fontes.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Recursos de cores

Com a utilização de CSS, podemos manipular as cores de elementos HTML, seja na aparência das caixas seja na cor de texto. Para isso, há uma série de propriedades CSS disponíveis para diversos elementos, mas antes vamos abordar as formas de definição de cores.

Formas de escrita de cores

As cores em CSS podem ser escritas de três modos:

- Com palavras-chave, nas quais podem ser usados os nomes das cores (seguindo as definidas pela especificação CSS) ou a notação hexadecimal. Por exemplo: blue, red, #FFFFFF etc.]
- Com um sistema de coordenada cúbica RGB, com as notações rgb() e rgba().
- Com um sistema de coordena cilíndrica HSL, com as notações hsl() e hsla().

Propriedades de cor

Essas propriedades se referem a quais elementos podemos definir cores.

Observe a seguir as quatro principais propriedades relacionadas à cor e aos elementos em que podem ser aplicadas.

color

Serve para definir: Cor de textos.

Onde pode ser utilizada: Elementos que contenham texto, como <h1> ... <h6>, <p>, <header>, <section>, etc.

background-color

Serve para definir: Cor de fundo de elementos.

Onde pode ser utilizada: Aplica-se a qualquer elemento HTML.

border-color

Serve para definir: Cor da borda.

Onde pode ser utilizada: Aplica-se a qualquer elemento HTML.

outline-color

Serve para definir: Cor da borda externa.

Onde pode ser utilizada: Aplica-se a qualquer elemento HTML.

Recursos de textos

A estilização de textos com o uso de CSS é dividida em duas partes:

- **Layout do texto:** Espaçamento entre os caracteres e linhas; alinhamento em relação ao container.
- **Estilos das fontes:** Família, tamanho, efeitos como negrito etc.

Em linhas gerais, os navegadores aplicam estilos padrões quando renderizam conteúdos textuais. Observe algumas propriedades CSS que alteram esse comportamento padrão.

Alinhamento de texto

A propriedade text-align é usada para controlar o alinhamento do texto em razão do container no qual está inserido.

Tal propriedade pode assumir quatro valores: left, right, center e justify. Como os nomes indicam, essas propriedades alinham o texto à esquerda, à direita, ao centro ou de forma justificada.

Espaçamento entre linhas

A propriedade line-height permite alterar o espaçamento vertical entre as linhas de texto. Seus valores possíveis são:

Normal

Valor padrão do navegador (entre 1 e


Número

Valor inteiro ou decimal que será multiplicado ao tamanho da fonte.

Comprimento

Valor de unidades, como pixels, pontos, etc.

A maneira mais recomendada para declarar o espaçamento entre linhas é utilizando o valor em número. Desse modo, o espaçamento será o resultado da multiplicação do valor definido pelo tamanho da fonte.



Exemplo

Line-height: 1.5; font-size: 12px. Sendo o valor 1.5 multiplicado pelo valor da propriedade font-size, resultando no valor de 18px de espaçamento.

Espaçamento entre letras e palavras

As propriedades letter-spacing e word-spacing permitem alterar o espaçamento entre letras e/ou palavras. Podem assumir valores de comprimento – “px”, “pt” etc.

Recursos de fontes

Em relação às fontes, há propriedades CSS para definir família, tamanho, estilo, entre outras possibilidades. Vamos conhecer as propriedades mais usadas!

Font-family

Essa propriedade é utilizada para definir a família da fonte utilizada em página web ou em partes de seu conteúdo. Utilizando essa propriedade, é possível definir, por exemplo, desde uma única fonte a uma lista de fontes, no qual seus valores são declarados em ordem de importância, da esquerda para direita. Desse modo, caso determinada fonte não esteja disponível no dispositivo cliente, a próxima fonte definida será usada, e assim sucessivamente. Caso nenhuma das fontes definidas esteja disponível no cliente, o navegador fará uso de uma fonte padrão.

Estes são exemplos de famílias de fonte: Arial, Verdana, Times New Roman (fontes com nomes compostos devem ser declaradas utilizando-se aspas), entre outras.

Essas fontes e algumas outras formam o conjunto chamado de **fontes seguras** para web (**Web Safe Fonts**).

Prefira usar fontes seguras sempre que possível, pois elas são suportadas pela maioria dos sistemas operacionais.

Verdana

Aa Ee Rr a

Aa Ee Rr a

abcdefghijklmnopqrstuvwxyz

1234567890

Exemplo de família de fonte (Verdana)

Font-size

Essa propriedade é responsável por definir o tamanho do texto. Seus valores podem ser definidos com a utilização de diferentes unidades de medida, como pixels, além de porcentagem etc.

Font-style

Propriedade usada na estilização de textos aplicando o efeito de itálico. Seus valores possíveis são: normal (ou seja, tira o efeito do texto, sendo o estilo padrão de todo elemento), italic e oblique (uma versão mais inclinada em relação ao itálico).

Font-weight

O peso de uma fonte é definido com a utilização dessa propriedade. Com ela, é possível aplicar o efeito de negrito em uma escala. Seus valores possíveis são: normal, bold, lighter e bolder (aumentam ou diminuem o peso da fonte em relação ao peso da fonte de seu elemento pai); e uma escala numérica de 100 a 900.



Atenção

Existem boas práticas e cuidados a serem considerados quando se trabalha com estilização de fontes usando CSS. Um desses cuidados diz respeito ao controle sobre a possível degradação que pode ocorrer na página. Portanto, tome os devidos cuidados optando pela utilização de uma lista de fontes e mantendo por último as fontes genéricas, como Serif, Sans Serif e Monospace. Desse modo, haverá maior garantia e controle sobre o que o usuário verá como resultado.

Web fontes

São um importante recurso em termos de tipografia. Se antes a sua estilização ficava restrita àquelas disponíveis nos sistemas operacionais dos usuários, a partir da implementação da regra @font-face tornou-se possível a utilização de web fontes. Essa nova propriedade permite a utilização de fontes que, ao serem definidas, são baixadas pelo navegador no momento de carregamento da página. Logo, sua utilização permite um controle maior do layout de uma página no que diz respeito às fontes, além da possibilidade de serem usadas fontes com maior apelo visual.

Como utilizar a regra @font-face

A declaração da regra @font-face é feita pela definição de duas principais propriedades:

font-family

Definimos um nome para a família da fonte que agora a fonte Awesome sendo declarada. Veja como estamos importando, usando o @font-face ao longo do arquivo CSS

```
@font-face {
  font-family: 'Awesome';
  font-style: normal;
  font-weight: normal;
  src: local('Awesome Font'),
       url('/assets/Fontes/awesome.woff2') format('woff2'),
       url('/siteondeafonteestadisponivel/Fontes/awesome.woff') format('woff'),
       url('/assets/Fontes/awesome.ttf') format('truetype'),
       url('/outrositeondeafonteestadisponivel/Fontes/awesome.eot') format('embedded-opentype');
}
```

Declaração de web fonte. Captura de tela do Notepad++

src

Aponta para a url na qual o arquivo da fonte se encontra.

Como podemos observar, além das propriedades font-family e src, há outras que podem ser aplicadas às web fontes. Em relação à font-family, a partir do momento da sua declaração, o nome definido poderá ser utilizado para estilizar qualquer outro elemento ao longo do CSS – considere que podemos tanto utilizar uma única família de fontes para o documento HTML inteiro como a combinação de diferentes famílias.

O código também mostra que as fontes incorporadas podem estar hospedadas localmente e na internet. Além disso, há outros elementos na declaração: as funções local e format. Vamos lá conferir!

Função local

Essa função instrui o navegador a verificar se a fonte definida está disponível na máquina do usuário antes de fazer o download.

Função format

Também chamada de dica, essa função opcional é utilizada quando se deseja declarar vários formatos de fontes, indicando justamente o formato de cada uma. No exemplo acima, temos os formatos “woff”, “woff2”, “ttf” e “eot”.

Tipos de web fontes

Atualmente existem diferentes tipos de web fontes. Ao escolher, considere a compatibilidade com a maioria dos navegadores e o tamanho dos arquivos. Vejamos os tipos mais comuns de web fontes!

	4.0	3.5	9.0	3.1	10.0
	5.0	3.6	9.0	5.1	11.1

	36.0	35.0	--	--	26.0
	4.0	--	--	3.2	9.0
	--	--	6.0	--	--

Tipos mais comuns de web fontes.
Alexandre Paixão

Como vimos, alguns tipos oferecem melhor suporte em relação aos navegadores mais atuais. Entretanto, não dão suporte às versões antigas. Isso reforça a recomendação anterior de usar sempre diferentes fontes para oferecer uma melhor experiência aos usuários.

Atividade

Sobre a estilização de textos e fontes, os navegadores possuem estilos padrões para esses tipos de elemento. Logo, é correto afirmar que:

☐

A

os estilos aplicados por padrão pelos navegadores existem para permitir que o controle do layout do conteúdo da página fique nas mãos do usuário e não do desenvolvedor.

☐

B

os navegadores padronizam os estilos dos elementos de texto e fonte para garantirem a usabilidade e acessibilidade das páginas.

☐

C

a CSS permite total controle sobre os elementos de texto e fonte. Com isso, todo o controle fica nas mãos do desenvolvedor, que poderá alterar qualquer aspecto desses elementos, tornando assim a página uniforme, uma vez que não dependerá dos estilos padrão dos navegadores, que são diferentes entre si.

☐

D

embora a CSS permita a estilização de textos e fontes, os navegadores sempre terão controle sobre o layout da página, podendo, inclusive, redefinirem os estilos CSS que não estejam de acordo com os padrões de acessibilidade.

☐

E

a CSS permite controle apenas parcial sobre os elementos de fonte, pois alguns navegadores padronizam os estilos desses elementos. Por outro lado, em relação aos elementos de texto, o controle é total e exclusivo dos desenvolvedores, independentemente do estilo padrão dos navegadores que venham a exibir a página.

☒

A alternativa C está correta.

A CSS permite total controle sobre qualquer elemento em uma página. Deve-se ter em mente, ao utilizá-la, não só as preocupações com estética, mas também com usabilidade e acessibilidade, garantindo assim a melhor experiência possível aos usuários.

Utilizando web fontes com CSS3

A versão 3 da CSS trouxe algumas novidades em relação às versões anteriores. Entre elas, a expansão das possibilidades de fontes a serem usadas numa página HTML. Nessa prática você aplicará o que vimos sobre Web Fontes, criando uma página HTML simples e a estilizando com tal recurso.

Assista ao vídeo e entenda como utilizar o recurso de web fontes, disponível apartir da especificação da CSS3.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Roteiro de prática

Você criará uma página HTML simples, composta por dois parágrafos de texto. A partir desses textos, e seguindo o roteiro abaixo, você deverá estilizá-los usando duas diferentes Web Fontes. Vale lembrar que, para a realização deste exercício, você precisará das seguintes ferramentas: um editor de textos – pode ser o próprio bloco de notas do Windows ou o Nano Editor do Linux, ou então algo um pouquinho mais avançado, como o software (livre/gratuito) Notepad++ e de um navegador – Google Chrome, Firefox, MS Edge etc.

Agora, vamos ao roteiro!

- Procure e baixe duas diferentes web fontes – tenha cuidado com os direitos de licença.
- Salve as web fontes (dê preferência aos arquivos .woff2 ou .woff) na mesma pasta em que for salvar a página HTML.
- Insira o código básico de uma página HTML e duas tags de parágrafo, com algum texto dentro.
- Atribua diferentes seletores para cada um dos blocos de parágrafo.
- Utilizando CSS, crie as propriedades @font-face para cada fonte que baixou.
- Atribua as propriedades acima para cada parágrafo de texto.
- Teste a página por meio do navegador.

Como resolução, teremos o código a seguir.

plain-text

Usando Web Fontes

Lorem ipsum dolor sit amet, consectetur adipiscing elit.
In euismod nibh sit amet justo cursus, eget posuere arcu pharetra.
Donec vestibulum, lectus non consectetur condimentum, ex purus
luctus lectus, at lacinia augue sem euismod diam.

Aliquam aliquam accumsan varius. Fusce facilisis sollicitudin tortor.
Curabitur aliquet bibendum ligula, vitae porta nunc ornare sed.
Aliquam dapibus scelerisque turpis, eget hendrerit turpis elementum in.
Aliquam accumsan sit amet nisi euismod imperdiet.

Faça você mesmo!

A partir da CSS3 podemos utilizar novas formas de escrita de cores, como HSL e HSLA, por exemplo. Sobre tais formas, analise as alternativas abaixo e sinalize a opção correta para a declaração de cor no formato HSL:

A

background-color:hsl(0, 0, 50%)



B

background-color:hsl(red, blue, white)



C

background-color:hsl('#ffcc00', 'ff0000','#000000')



D

background-color:hsl(0, 100%, 50%)



E

background-color:hsl(0, 100%, blue)



A alternativa D está correta.

A declaração de cor utilizando o formato HSL precisa ser feita passando 3 valores dentro de uma escala, sendo o primeiro um número inteiro, entre 0 e 360; e os dois seguintes percentuais entre 0% e 100%.

3. Conceitos avançados de CSS

Conceitos avançados

Agora que já vimos os conceitos básicos de CSS, sua sintaxe, seus elementos e suas formas de integração com HTML, vamos analisar os conceitos avançados, como o box model (modelo de caixas ou retângulos).

Neste vídeo, abordamos os conceitos avançados de CSS, como box-model, pseudoclasses, pseudoelementos, posicionamento e layout.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Conceitos de box model

Podemos comparar os elementos de uma página web, quando pensamos na organização de seus elementos e marcação, a brinquedos de montar do estilo blocos. Vejamos!



Montagem

Nos brinquedos de bloco há peças de diferentes tamanhos, largas, alturas, cores e formatos. Tendo em mãos tais elementos, é nosso papel montá-los, encaixá-los de forma harmoniosa para, ao final, obtermos o resultado esperado.



Resultado

Para chegar a esse resultado, é importante entendermos o comportamento, a composição e as características de cada bloco, assim como os estilos que podem receber.

Nesse sentido, dentro do conceito de box model – que, em CSS, está relacionado a design e layout – nossos boxes possuem quatro componentes principais: margem (margin), borda (border), preenchimento (padding) e conteúdo (content).

```
1 <!doctype html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>Box Model em CSS</title>
6   <style type="text/css">
7     div{color:#fff; font-weight:bold;font-size:20px;}
8     #box_inicial{
9       width:400px; height:200px;
10      background-color:red;
11    }
12
13    #box_exemplo{
14      width:400px; height:200px;
15      background-color:blue;
16      margin-top:30px;
17      margin-bottom:50px;
18      margin-left:15px;
19      margin-right:20px;
20      padding:50px;
21      border: 10px solid black;
22    }
23
24    #box_final{
25      width:400px; height:200px;
26      background-color:green;
27    }
28  </style>
29 </head>
30 <body>
31   <div id="box_inicial">
32     Conteúdo do box inicial.
33   </div>
34   <div id="box_exemplo">
35     Conteúdo do box de exemplo.
36   </div>
37   <div id="box_final">
38     Conteúdo do box final.
39   </div>
40 </body>
41 </html>
```

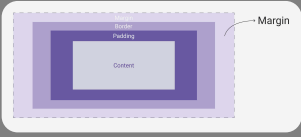


Componentes do box model

Repare que na imagem anterior foram definidas 3 caixas com o elemento <div> e, para cada uma delas, foram definidos diferentes estilos, como largura, altura e cor de fundo. Para a <div> com identificador “box_exemplo” foram declarados ainda valores para margin, padding, border e content. Veja a explicação desses valores:

Margin

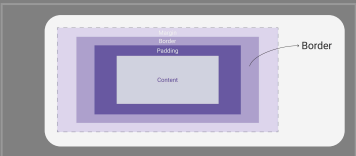
Como indicado no lado direito da imagem anterior, a margem permitiu que um espaço fosse criado entre a primeira, a segunda e a última div. Também criou um espaçamento entre a box exemplo e a lateral esquerda da página. As margens de um elemento podem ser controladas com as propriedades CSS margin-top, margin-bottom, margin-right e margin-left – além do atalho margin, como visto no código de exemplo.



Border

A borda define a área do elemento, incluindo altura e largura, e também permite a definição de uma cor diferente para essas extensões do elemento. Ela é controlada pela propriedade CSS "border" e suas variantes, que permitem definir a largura, a cor e o tipo de borda.

O tamanho declarado para a borda é somado ao tamanho declarado para o elemento, compondo, assim, o seu tamanho final.



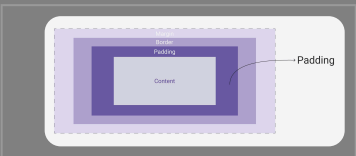
Padding

Para entender a função do padding, repare que os textos da primeira e da última <div> estão colados no topo e na lateral esquerda. Entretanto, na div do meio, há um espaçamento em relação ao topo e à lateral esquerda. Esse espaço de preenchimento equivale ao padding. Assim como a margem, suas dimensões são a altura e largura.

Atente-se para a seguinte diferença: margin diz respeito **ao espaço entre elementos**; já o padding refere-se **ao conteúdo interno do próprio elemento**, além de fazer parte dele.

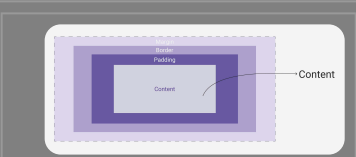
Na prática, a diferença entre margin e padding significa que a div #box_exemplo, cuja largura foi declarada como 400px e altura como 200px tem, na verdade, 500px de altura e 300px de largura. Ou seja, o padding aumentou suas dimensões: width + padding-right + padding-left ⇒ 400px + 50px + 50px = 500px height + padding-top + padding-bottom ⇒ 200px + 50px + 50px = 300px.

Para controlar o preenchimento de um elemento, são utilizadas as propriedades CSS padding-top, padding-bottom, padding-right e padding-left, além do atalho padding.



Content

Essa é a área interna do elemento, ocupada pelo seu conteúdo. Suas dimensões são altura e largura. Além disso, sua cor de fundo (background), a cor da fonte (color) de seu conteúdo, sua largura (width, min-width, max-width) e altura (height, min-height, max-height) podem ser estilizadas com CSS.



Conceitos de pseudoclasses e pseudoelementos

Uma declaração CSS é composta pelo elemento que se deseja estilizar, pela propriedade a ser estilizada e pelo valor a ser atribuído. Além disso, vimos que o elemento pode ser definido de maneira ampla (utilizando-se o nome da tag), específica (pelo seu identificador único) e seletiva (com a utilização de classes).

Um elemento filho pode, ainda, herdar as propriedades de um elemento pai. Todos esses modos de definir estilo são bastante abrangentes. Entretanto, existem algumas formas especiais e muito úteis para se aplicar estilos: as pseudoclasses e os pseudoelementos.

Pseudoclasses

São utilizadas para definir um estado especial de um elemento. Por exemplo, podemos mudar o estilo de um elemento ao passarmos o mouse sobre ele (evento mouseover). Esse novo estilo é temporário, ou seja, não corresponde ao seu estado natural. Também podemos mudar o estilo de um link que foi clicado, alterando sua cor ou alguma outra propriedade.

A sintaxe para declaração da pseudoclasse é composta pela palavra-chave correspondente ao nome da pseudoclasse precedido pelo sinal de dois pontos.



Exemplo

```
div:hover{background-color:#000000;}
```

Veja a seguir a lista básica de cinco pseudoclasses.

:active

Como declarar: a:active

Para que serve: Selecionar todos os links ativos.

:checked

Como declarar: input:checked

Para que serve: Selecionar todos os campos input checados.

:first-child

Como declarar: li:first-child

Para que serve: Selecionar todo primeiro item de lista.

:last-child

Como declarar: li:last-child

Para que serve: Selecionar todo último item de lista.


:hover

Como declarar: div:hover

Para que serve: Selecionar todas as divs no evento mouseover.

Pseudoelementos

São palavras-chave que podem ser adicionadas ou relacionadas a um seletor para estilizar uma parte específica dele. A imagem a seguir mostra duas declarações CSS, uma sem e outra com o uso de pseudoelemento. Em ambas, é definido que a primeira letra de texto em um parágrafo tenha tamanho e cor diferentes do restante do texto.



Conteúdo interativo

Acesse a versão digital para ver mais detalhes da imagem abaixo.


```
<!DOCTYPE html>
<html lang="pt-BR">
  <head>
    <meta charset="utf-8">
    <title>Box Modal em CSS</title>
    <style type="text/css">
      p#sem_pseudo_classe{
        font-size: 12px;
        color: #000000;
      }
      p#sem_pseudo_classe span.primeira_letra_fonte_maior{
        font-size: 26px;
        color: #0000ff;
      }
      p#com_pseudo_classe{
        font-size: 12px;
        color: #000000;
      }
      p#com_pseudo_classe:first-letter{
        font-size: 26px;
        color: #0000ff;
      }
    </style>
  </head>
  <body>
    <p id="sem_pseudo_classe">
      <span class="primeira_letra_fonte_maior">T</span>exto do parágrafo sem pseudo classe.
    </p>
    <p id="com_pseudo_classe">
      Texto do parágrafo com pseudo classe.
    </p>
  </body>
</html>
```

Texto do parágrafo sem pseudo classe.

Texto do parágrafo com pseudo classe.

Exemplo de utilização de pseudoelementos

Ao analisar, codificar e testar o código acima, você perceberá que, no primeiro parágrafo, foi necessário utilizar um elemento a mais, a tag ``, ao redor da primeira letra do texto para poder estilizá-la. Já no segundo parágrafo, o mesmo estilo foi alcançado apenas com o uso do pseudoelemento `first-letter`. A utilização do pseudoelemento diminui a quantidade de código, tornando sua compreensão mais clara.

Cabe destacar outro ponto do exemplo relacionado à sintaxe dos pseudoelementos: neles são usados dois pontos duplos (ou dobrados) para a declaração. Esse uso proposital é para diferenciá-los das pseudoclasses.

Veja a seguir os cinco pseudoelementos principais.

::after

Exemplo: `img::after`

Para que serve: Inserir conteúdo após o elemento indicado.

::before

Exemplo: `h1::before`

Para que serve: Inserir conteúdo antes do elemento indicado.

::first::letter

Exemplo: `p::first-letter`

Para que serve: Selecionar a primeira letra do elemento indicado.

::first-line

Exemplo: `p::first-line`

Para que serve: Selecionar a primeira linha do elemento indicado.

::selection

Exemplo: `p::selection`

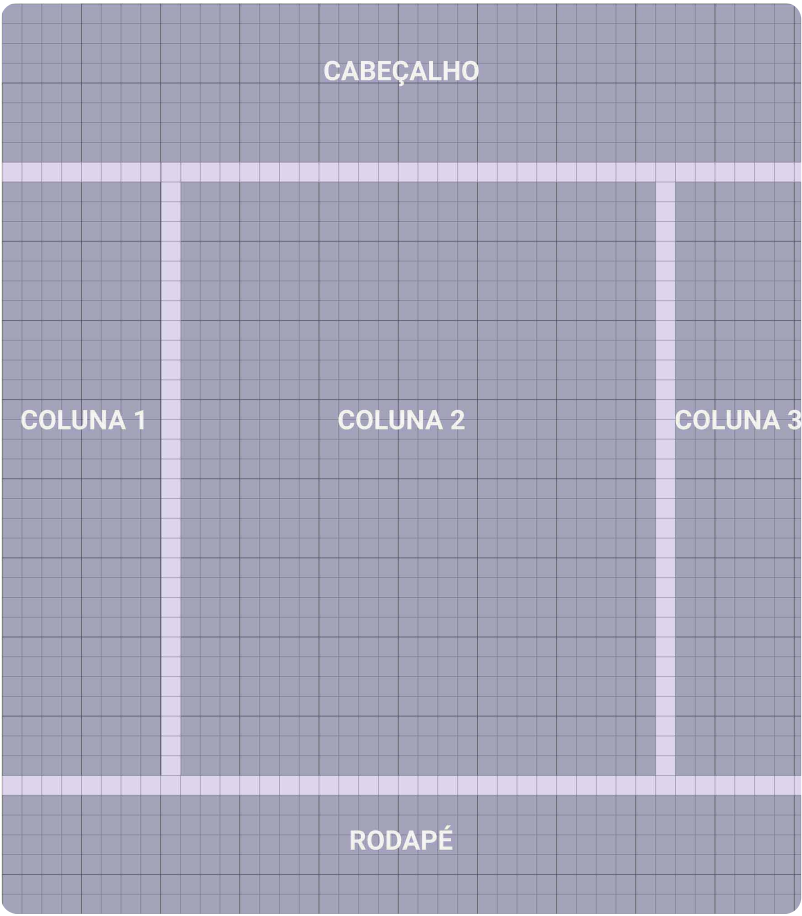
Para que serve: Selecionar a porção de um elemento que é selecionado pelo usuário

Conceitos de posicionamento e layout

Para lidar com posicionamento em CSS, é necessário revisitar alguns conceitos já abordados, principalmente aqueles relacionados ao box model. Além disso, é importante lembrar que os elementos HTML possuem estilos e comportamentos padrão. Alguns desses padrões diferem um pouco de um navegador para outro e podem ser modificados por CSS, onde as tais diferenças são resolvidas e, de fato, um padrão de comportamento pode ser definido.

Layout em colunas e grid layout

Esses dois conceitos são importantes quando tratamos da estrutura visual de páginas HTML. Em uma definição simplista, ambos tratam de como os elementos boxes podem ser posicionados e organizados em uma página. Veja a seguir a estrutura em colunas de uma página:



Layout CSS em colunas

Em termos de HTML, apenas utilizando boxes (header, footer, section, aside, nav, div etc.), os elementos ficariam empilhados uns sobre os outros. Para aplicar o layout visto na imagem e posicionar os elementos na página, precisaremos utilizar CSS.

Propriedade position

É a propriedade CSS responsável pelo posicionamento. Seus valores possíveis são: static, relative, fixed, absolute e sticky. Além disso, as propriedades top, bottom, right e left são usadas em conjunto, a fim de definir os valores das respectivas distâncias e, consequentemente, do posicionamento. Tais propriedades, inclusive, só podem ser usadas quando for definido um valor para position.

Conheça agora cada propriedade position!

Position static

É a posição padrão dos elementos. Desse modo, elementos definidos como static ou sem a propriedade position são posicionados naturalmente, de acordo com o fluxo normal da página, não assumindo nenhuma localização especial. Inclusive, as propriedades top, bottom, right e left não são refletidas em elementos estáticos.

Position relative

Faz com que um elemento seja alocado de modo relativo à sua posição normal. Com isso, ao definirmos valores para as propriedades top, bottom, right e left, ajustamos a sua localização em relação à sua posição natural.

Position fixed

É utilizado quando desejamos definir uma posição fixa para um elemento na página. Com isso, independentemente do scroll, de rolarmos a página para cima ou para baixo, o elemento sempre permanecerá no mesmo local. As propriedades top, bottom, right e left devem ser usadas para definir o lugar no qual o elemento será fixado. Esse elemento é posicionado em relação à viewport/janela do navegador. Com isso, ele “flutuará” sobre os demais conteúdos da página, ficando fixo onde foi colocado e não ocupando, assim, a posição original na qual foi declarado no HTML.

Position absolute

Permite que um elemento seja posicionado em relação à localização do seu elemento ancestral mais próximo – o qual também deverá estar posicionado, ou seja, não poderá ser static. Quando o elemento definido como absolute for o primeiro elemento da página, ele então será posicionado em relação ao . Com isso, tal elemento acompanhará a rolagem da página.

Position sticky

Permite que um elemento seja posicionado com base na posição de rolagem da página (scroll). Com isso, seu comportamento varia entre o relativo e o fixado, dependendo da posição do scroll. Essa propriedade é mais recente em termos de especificação e não possui suporte em todas as versões dos navegadores. É usada, normalmente, quando queremos criar um efeito de sobreposição de conteúdo. Na prática, o elemento é visualizado ao abirmos uma página. Ao rolarmos para baixo, ele se mantém fixo, com os demais conteúdos passando sob ele.

Atividade

Sobre as propriedades e dimensões do box model de um elemento <div>, com os estilos definidos abaixo, indique a afirmação correta.

div{

width:500px!important;
border: 5px solid black;
padding-top: 10px;
padding-right:10px;
padding-bottom: 5px;
margin-left:50px
}

A

A largura final da div será de 500px.

B

A largura final da div será de 520px.

C

A largura final da div será de 510px.

D

A largura final da div será de 570px.

E

A largura final da div será de 590px.


✓

A alternativa B está correta.

As dimensões de largura e altura são alteradas de acordo com a borda e o padding definidos. No exemplo temos: 500px + 5px (borda da direita) + 5px (borda da esquerda) + 10px (padding da direita) = 520px.

Alterando o posicionamento de elementos utilizando CSS

Neste vídeo, demonstramos como alterar o posicionamento de elementos em uma página HTML utilizando CSS.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Roteiro de prática

Você aplicará os conceitos de posicionamento e layout para modificar o layout inicial de uma página, organizando seus elementos e definindo novas propriedades. Você verá a seguir o “antes” e o “depois”, assim como o código-fonte inicial.

Mas antes de começar, vamos revisar as ferramentas e também o roteiro da prática. Primeiro, as ferramentas: um editor de textos – pode ser o próprio bloco de notas do Windows ou o Nano Editor do Linux, ou então algo um pouquinho mais avançado, como o software (livre/gratuito) Notepad++ e também de um navegador – Google Chrome, Firefox, MS Edge etc.

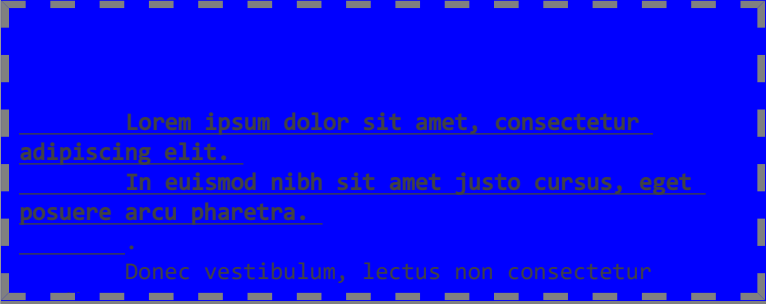
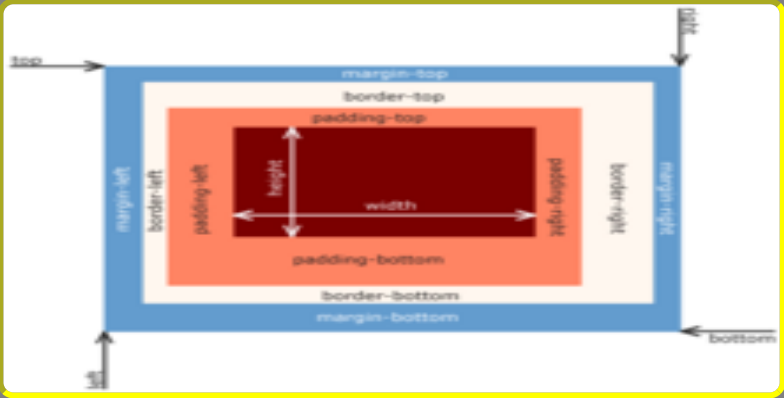
Agora, vamos ao roteiro!

- Copie o código disponibilizado a seguir e o salve, utilizando seu editor, com a extensão “.html”.
- Criando uma página com o mesmo código ou apenas alterando esse inicial, modifique os elementos HTML e propriedades CSS para que a página fique com o layout desejado (imagem a seguir).
- Salve todo o seu trabalho e teste a página do navegador.

Código HTML inicial:

plain-text

Visão Inicial da Página



Veja como estará a página antes da estilização pedida no roteiro:

Posicionamento elementos com x

Arquivo

Visão Inicial da Página



Página antes da estilização

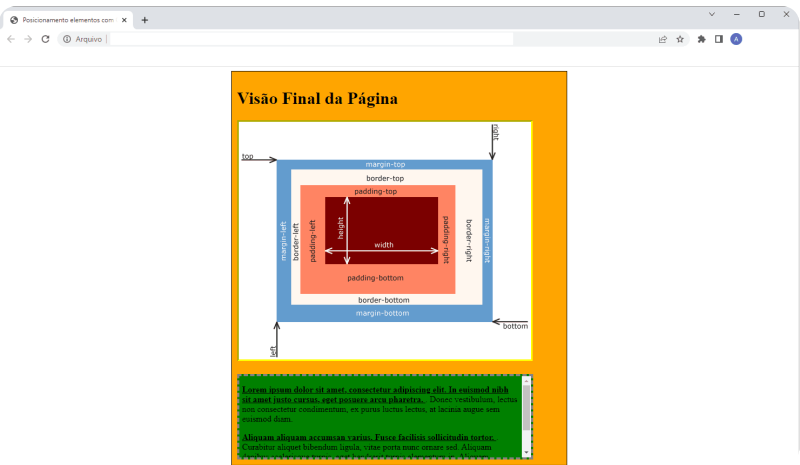
Como resolução, teremos o código a seguir.

plain-text

Visão Final da Página



Veja como ficará a página depois da estilização pedida no roteiro:



Página depois da estilização

Faça você mesmo!

Cada elemento possui uma posição natural, que pode variar entre navegadores. Para garantir consistência na exibição dos elementos em diferentes navegadores, é necessário utilizar propriedades CSS. Considere o fragmento de código abaixo, em que a tag <p> possui o valor "relative" atribuído a ela com a respectiva propriedade CSS. A seguir, escolha a alternativa correta.

plain-text

...

Texto

...

```
p{
  position:relative;
}
```

☐ A

A tag <p> será posicionada de forma relativa em relação ao seu elemento ancestral, ou seja, em relação a <div>.

☐ B

A tag <p> será posicionada em função da tag <body>, uma vez que não foi declarada uma propriedade position para a <div>.

☐ C

A tag <p> será posicionada da mesma forma se nenhuma propriedade de posicionamento lhe fosse atribuída.

☐ D

Para assumir a posição relativa, a tag <p> precisaria estar localizada fora da <div> ou de qualquer outro elemento pai.

☐ E

Nenhum estilo de posicionamento será aplicado à tag <p>, já que ela não recebeu nenhum seletor (classe ou id).

☒

A alternativa C está correta.

As propriedades de posicionamento precisam ser utilizadas em conjunto com as propriedades top, bottom, right e left – e seus respectivos valores. Do contrário, nenhuma mudança será aplicada ao seu posicionamento. No código dessa atividade, a declaração CSS será ignorada pelo navegador.

4. Frameworks CSS

Frameworks e o desenvolvimento web

O uso de frameworks no processo de desenvolvimento de software não é algo novo. Você encontrará várias discussões apoiando ou desestimulando o seu uso. Independentemente da sua linha de pensamento, é importante saber que tais frameworks também estão disponíveis no desenvolvimento web, incluindo na estilização de páginas web via CSS.

Neste vídeo, vamos apresentar o conceito de frameworks e o uso desse recurso no desenvolvimento web.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Frameworks e CSS

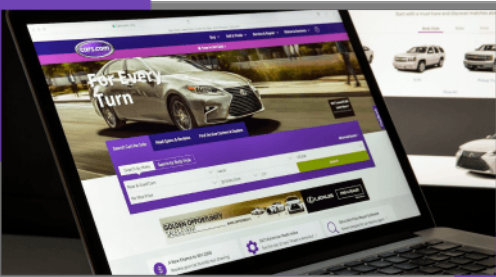
A CSS é uma tecnologia poderosa, flexível e, muitas vezes, complexa. São várias propriedades e valores possíveis. Inúmeras combinações podem se sobrepor umas às outras, inclusive.

A marcação HTML tem um comportamento natural em relação aos elementos, além de pequenas variações entre navegadores. Por outro lado, há bastante similaridade em relação aos layouts de diversos sites.

Os sites de e-commerce, por exemplo, costumam ter um layout bem parecido para facilitar a experiência do usuário ao trafegar entre um e outro.

Nesses casos, vale o ditado: “não é necessário reinventar a roda”; aliás, um bom ponto de partida para falarmos sobre frameworks CSS.

A CSS é uma tecnologia poderosa, flexível e, muitas vezes, complexa. São várias propriedades e valores possíveis. Inúmeras combinações podem se sobrepor umas às outras, inclusive.



Exemplo de layout de site

A marcação HTML tem um comportamento natural em relação aos elementos, além de pequenas variações entre navegadores. Por outro lado, há bastante similaridade em relação aos layouts de diversos sites. Os sites de e-commerce, por exemplo, costumam ter um layout bem parecido para facilitar a experiência do usuário ao trafegar entre um e outro. Nesses casos, vale o ditado: “não é necessário reinventar a roda”; aliás, um bom ponto de partida para falarmos sobre frameworks CSS.

Frameworks podem ser definidos como um conjunto de componentes reutilizáveis que permitem a otimização do processo de programação.

A maioria dos frameworks CSS mantém similaridades entre si, além de prós e contras específicos, que vão desde a facilidade de aprendizagem ao suporte e à documentação disponíveis, entre outros fatores.

Logo, a escolha de um framework pode se dar por fatores objetivos, relacionados às suas características ou aos requisitos específicos de determinado projeto, ou mesmo por fatores subjetivos, como gosto pessoal. Ao decidir utilizar um framework, é fundamental ter em mente o quanto ele poderá auxiliá-lo em seu trabalho. Para isso, é importante conhecer suas principais características, vantagens e desvantagens de cada opção.

Fique agora com **Bootstrap**, **Foundation** e **Semantic UI**, os três principais frameworks CSS existentes!

Bootstrap

Desenvolvido pela equipe do Twitter em 2011, posteriormente passou a ser mantido de modo independente. Sua licença é open source e, atualmente, é o framework CSS mais popular.

Trata-se de um framework responsivo baseado na premissa mobile-first – cujo foco inicial são os dispositivos móveis e, em seguida, os desktops. Possui componentes prontos para uso (ready-to-use) desenvolvidos em HTML, CSS e Javascript.

Sistema de grid

O Grid Layout é um sistema de layout generalizado. Com ênfase em linhas e colunas, pode parecer um retorno ao layout da tabela, mas há muito mais no layout da grade do que no layout da tabela (MEYER, 2017).

Uma das principais características dos frameworks CSS é o seu **sistema de grid**. Veja agora a difença entre grid e sistema grid:

Grid

É um elemento de design, uma ferramenta que serve para ordenar elementos visuais.

Sistema de grid

Consiste em uma série de containers, linhas e colunas que servem para arranjar e alinhar conteúdo.

Embora compartilhem da mesma fundamentação teórica, há pequenas diferenças de implementação entre os frameworks.

A grid do Bootstrap, por exemplo, possui 12 colunas e 5 breakpoints responsivos, que são pontos de quebra nos quais o layout será ajustado para atender a diferentes resoluções de tela. Esses breakpoints são:

- Extra small
- Small
- Medium
- Large
- Extra large

Na prática, esse sistema deve ser corretamente utilizado para que todos os elementos da página sejam alinhados e visualizados em diferentes tamanhos de tela.

	None (auto)	540px	720px	960px	1140px
	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
	12				
	30px (15px on each side of a column)				
	Yes				
	Yes				


Sistema de grid do Bootstrap
Bootstrap Grid System, 2023

Como utilizar o Bootstrap


É necessário incluir a sua biblioteca, composta por dois arquivos: um **CSS** e outro **Javascript**. Essa instalação é simples e pode ser feita pela inclusão dos respectivos arquivos diretamente na HTML.

Outra forma de instalação é por meio de ferramentas gerenciadoras de pacotes, como **npm** ou **gem**. Em termos de dependência, para a utilização completa de todas as suas funcionalidades, é necessário ainda incluir outras bibliotecas **Javascript**, a **Jquery** e a **Popper**.

Por fim, é importante considerar a compatibilidade das versões do framework, tanto em relação a bibliotecas de terceiros quanto às funcionalidades que possam estar obsoletas ou depreciadas.



Ferramenta Jquery



Comentário

O Bootstrap possui inúmeras classes predefinidas para as mais diversas necessidades. Para utilizá-las, é preciso combiná-las com uma marcação HTML predefinida, conforme documentação oficial. Por exemplo: há uma classe que pode ser aplicada em tabelas para criar o efeito zebra (alternância de cores entre as linhas da tabela), a “table-striped”. Para usar essa classe, basta incluir seu nome no atributo class de uma tabela.

Foundation

Framework criado em 2011 e que está entre os mais conhecidos e utilizados. É responsivo e baseado na abordagem mobile-first. Sua principal característica é fazer uso nativo do pré-processador de CSS, chamado de SASS.


Sistema de grid

Sistema do Foundation também composto por **12 colunas**. Nas versões mais recentes, o sistema básico de grid foi substituído por um novo sistema, o **XY grid**. novas funcionalidades foram adicionadas, como alinhamento vertical e horizontal, dimensionamento automático e grid vertical completa.

Como utilizar o Foundation

É semelhante à do Bootstrap: é preciso incluir um **arquivo CSS** e outro **Javascript** ou então utilizar um gerenciador de pacotes. Além disso, é recomendado também incluir a biblioteca **jQuery**.

A respeito da compatibilidade, lembre-se, algumas funcionalidades são descontinuadas entre uma versão ou outra. Logo, tome cuidado para que nada deixe de funcionar ao atualizar versões.



Comentário

Em termos de recursos extras, destaca-se nesse framework a existência de recursos específicos para a criação de HTML responsivo para e-mail. Trata-se do Foundation for Emails.

Semantic UI

O Semantic UI se destaca por utilizar, nativamente, um pré-processador CSS, o LESS, e a biblioteca Javascript JQuery. Também é um framework open source. Suas classes utilizam sintaxe de linguagem natural, como substantivos, por exemplo, para vincular conceitos de maneira mais intuitiva.

Como utilizar o Semantic UI


A sua inclusão é semelhante à dos demais frameworks, ou seja, por meio de **arquivos CSS** e **JS**, além da biblioteca **jQuery**, ou via gerenciadores de pacotes.

Outros frameworks

Há vários outros disponíveis, sendo os mais conhecidos os seguintes:

<h3>Pure</h3> <p>Considerado o framework mais leve, foi</p>	<h3>Materialize CSS</h3> <p>Baseado no Material</p>
<h3>Bulma</h3> <p>Framework baseado unicamente em CSS,</p>	<h3>Skeleton</h3> <p>Framework minimalista. Possui apenas</p>

Como não faz uso em suas definições e em seus exemplos de utilização, os frameworks têm a mesma finalidade, servindo para as mesmas funções. Em outras palavras, tudo que é possível criar com um framework também é possível criar com outro. Porém há prós e contras em cada um deles, como tamanho do framework e seus impactos no carregamento da página, facilidade ou dificuldade de aprendizagem e simplicidade de sintaxes.



Recomendação

Na prática, é recomendado usar um framework sempre que possível. Escolha aquele que melhor atenda às suas necessidades. Na dúvida, escolha o mais utilizado – afinal de contas, ele não deve ser o mais usado à toa.

Vantagens e desvantagens da utilização de frameworks

Não existe uma recomendação definitiva sobre usar ou não um framework em um projeto web. Se, por um lado, a utilização de frameworks ajuda a otimizar o tempo de desenvolvimento, por outro lado, tira um pouco do controle do programador sobre o que está sendo feito, uma vez que não é recomendado alterar o comportamento padrão dos códigos fornecidos pelos frameworks.

Vantagens

- Padronização do código: muito válido, sobretudo, quando se trabalha em equipe.
- Economia de tempo: uma vez que não é preciso criar todo o código CSS do zero.
- Seguimento de padrões: já que os frameworks estão sempre antenados às especificações e recomendações oficiais.
- Compatibilidade entre navegadores: funcionam em diferentes navegadores.

Desvantagens

- Tamanho/peso do framework: Pode impactar no carregamento da página.
- Restrições de design: Lembre-se de que o framework possui um layout padrão, baseado em grids. Isso pode acabar limitando a imaginação no momento de criação do design ou impactando no tempo de desenvolvimento para que seja possível encaixar o layout criado no padrão estabelecido pelo framework.
- Curva de aprendizado: aprender a utilizar adequadamente um framework pode levar algum tempo.
- Controle sobre o código: sendo obrigado a utilizar a estrutura definida pelo framework, acabamos por perder o controle total sobre o código. Além disso, utilizar frameworks sem uma boa base teórica sobre CSS pode limitar o seu entendimento e aprendizado.

Atividade

Em relação à utilização de frameworks, assinale a afirmativa incorreta.

☐ A

Qualquer componente ou estilo disponibilizados pelos frameworks podem ser produzidos apenas com código CSS e Javascript, ou seja, sem a utilização de frameworks.

☐ B

Os frameworks são um importante recurso que auxiliam no desenvolvimento, diminuindo o tempo, padronizando o código e garantindo uma maior compatibilidade entre navegadores e dispositivos.

☐ C

Para um melhor resultado é importante utilizar vários frameworks em um mesmo projeto. Com isso, é possível aproveitar o que cada um oferece de melhor.

D

Não há um melhor ou um pior framework. Cada um oferece vantagens e desvantagens, prós e contras. Inclusive, alguns podem ser a melhor opção para determinado projeto e para outro não.

E

A escolha do framework a ser utilizado é complexa, pois eles são bem distintos. Apenas uma minoria dos frameworks CSS mantém similaridades entre si.

✓


A alternativa C está correta.

A utilização de vários frameworks CSS em um mesmo projeto pode causar inúmeros problemas, entre eles, conflitos de estilos, uma vez que alguns compartilham entre si os mesmos nomes de seletores. Logo, é imprescindível utilizar apenas um framework por projeto.

Aplicando o sistema de grid do Bootstrap

Como já vimos, o Bootstrap fornece, dentro de sua grid, um conjunto de containers + linhas + colunas que nos permite organizar e alinhar os elementos e, conseqüentemente, o conteúdo de uma página web. Siga o próximo roteiro para testar seus conhecimentos!

Neste vídeo, demonstramos a aplicação do sistema de grid do Bootstrap.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Roteiro de prática

Você usará o framework Bootstrap e seu Sistema de Grids para organizar os elementos de uma página, alinhando, com o uso de containers, linhas e colunas, ou seja, o conteúdo (siga a disposição mostrada na próxima imagem). Para a realização desse exercício, você precisará das seguintes ferramentas: um editor de textos – pode ser o próprio bloco de notas do Windows ou o Nano Editor do Linux, ou então algo um pouquinho mais avançado, como o software (livre/gratuito) Notepad++ – e também de um navegador (Google Chrome, Firefox, MS Edge etc).

Agora, vamos ao roteiro!

- Observe a disposição do conteúdo na imagem.
- A partir do seu editor, crie a estrutura básica de uma página HTML e importe o framework CSS.
- Dentro da tag body, crie a estrutura de <div> ou outro elemento que preferir, aplicando as classes do Bootstrap necessárias para que, ao exibir a página, a sua disposição visual fique como na imagem.
- Salve todo o seu trabalho e veja o resultado no navegador.

Como resolução, teremos o código a seguir.



Ao final, a página web deverá ser visualizada desta forma:



Resultado final da página da web

Faça você mesmo!

O sistema de grid dos frameworks CSS possui algumas semelhanças, como a organização do layout com o uso de containers, linhas e colunas. Nesse sentido, analise a seguinte imagem:

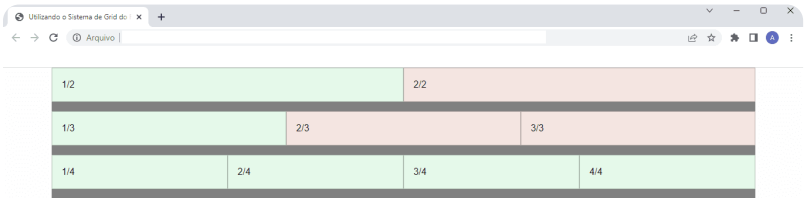


Imagem do exercício

Na imagem podemos ver a exibição de uma grid construída com o framework Foundation. A estrutura inicial dessa grid é composta por uma div (à qual foi atribuída a classe “row”), seguida por outra grid – aninhada à primeira (à qual foram atribuídas as classes large-12 e columns). Para chegar no resultado acima, onde temos uma sequência de div's aninhadas à primeira div (row), qual o conjunto de classes, apresentados nas alternativas a seguir, deveremos utilizar em cada div?

A

- large-12 columns ; large-12 columns
- large-6 columns ; large-4 columns ; large-2 columns
- large-4 columns ; large-4 columns; large-2 columns ; large-2 columns

B

- large-6 columns ; large-6 columns
- large-4 columns ; large-4 columns ; large-4 columns
- large-3 columns ; large-3 columns; large-3 columns ; large-3 columns

C

large-6 columns ; large-6 columns

large-3 columns ; large-3 columns ; large-3 columns ; large-3 columns

large-4 columns ; large-4 columns; large-4 columns

D

large-8 columns ; large-4 columns

large-3 columns ; large-3 columns ; large-3 columns ; large-3 columns

large-4 columns ; large-4 columns; large-4 columns

E

large-4 columns ; large-8 columns

large-4 columns ; large-4 columns ; large-4 columns

large-2 columns ; large-2 columns; large-2 columns ; large-6 columns



A alternativa B está correta.

A grid do framework Foundation possui no máximo 12 colunas. Para obter uma divisão igualitária entre elementos filhos dentro de um elemento pai definido como 12 (no nosso caso, large-12), os elementos filhos devem ser múltiplos de 12, dependendo do número de colunas desejado. Em nosso exemplo, a primeira linha possui duas colunas. Consequentemente, dois elementos anotados com "large-6" (6 + 6 = 12). A segunda linha, três colunas – 3 elementos anotados com "large-4" (4 + 4 + 4 = 12). Por fim, a última linha possui 4 colunas – 4 elementos anotados com "large-3" (3 + 3 + 3 + 3 = 12).

5. Conclusão

Considerações finais

O que você aprendeu neste conteúdo?

- Visão geral sobre a linguagem de marcação / folhas de estilos CSS
- Seletores e propriedades CSS
- Como usar CSS para estilizar cores, textos e fontes
- Conceitos de box model.
- O que são as pseudoclasses e os pseudoelementos em CSS.
- Conceitos de layout em uma página web.
- Alguns frameworks CSS, como Bootstrap e Foundation.

Explore +

Confira as indicações que separamos especialmente para você!

A página W3Schools disponibiliza inúmeros tutoriais sobre os mais diversos assuntos relacionados ao desenvolvimento web. Busque por:

- **CSS Pseudoclasses**
- **Google Fonts**
- **CSS RGB Color**
- **CSS HSL Color**

Indicamos também a **Google Fonts**, uma galeria de web fontes da Google com suporte em todos os navegadores.

Mozilla também disponibiliza diversos tutoriais e artigos de suporte ao desenvolvimento web. Leia:

- **Fundamental text and font styling**
- **CSS Media Queries**

Se quiser praticar a elaboração de códigos e visualizar a renderização no navegador, indicamos dois recursos on-line: **Codepen** e **JSFiddle**.

Referências

MEYER, E.; WEYL, E. CSS: **The Definitive Guide**. [s.l.]: O'Reilly Media, Inc., 2017.