**1. O que significa CSS e qual é o seu propósito principal?**

CSS é a linguagem que usamos para estilizar um documento HTML.

CSS descreve como os elementos HTML devem ser exibidos.

**2. Qual a diferença entre CSS inline, interno e externo?**

O CSS Inline serve para dar estilo a um elemento HTML específico. O CSS interno requer que você adicione a tag <style> no seu documento HTML, sendo efetivo para dar estilo a uma única página. Já o CSS externo opera linkando suas páginas a um arquivo .

**3. Explique o que é um seletor em CSS e dê três exemplos de seletores diferentes.**

Um seletor CSS seleciona o(s) elemento(s) HTML que você deseja estilizar.

Podemos dividir os seletores CSS em cinco categorias:

* Seletores simples (selecionam elementos com base no nome, id, classe)
* [Seletores combinadores](https://www.w3schools.com/css/css_combinators.asp) (selecionam elementos com base em um relacionamento específico entre eles)
* [Seletores de pseudoclasse](https://www.w3schools.com/css/css_pseudo_classes.asp) (selecionam elementos com base em um determinado estado)
* [Seletores de pseudoelementos](https://www.w3schools.com/css/css_pseudo_elements.asp) (selecionam e estilizam uma parte de um elemento)
* [Seletores de atributos](https://www.w3schools.com/css/css_attribute_selectors.asp) (selecionam elementos com base em um atributo ou valor de atributo)

Esta página explicará os seletores CSS mais básicos.

**4. O que é o modelo de caixa (Box Model) em CSS? Quais são as partes principais que o**

**compõem?**

Todos os elementos HTML podem ser considerados caixas.

Em CSS, o termo "modelo de caixa" é usado quando se fala em design e layout.

O modelo de caixa CSS é essencialmente uma caixa que envolve cada elemento HTML. Ele consiste em: conteúdo, preenchimento, bordas e margens. A imagem abaixo ilustra o modelo de caixa:

Explicação das diferentes partes:

* **Conteúdo** - O conteúdo da caixa, onde o texto e as imagens aparecem
* **Padding** - Limpa uma área ao redor do conteúdo. O padding é transparente
* **Borda** - Uma borda que circunda o preenchimento e o conteúdo
* **Margem** - Limpa uma área fora da borda. A margem é transparente

O modelo de caixa nos permite adicionar uma borda ao redor dos elementos e definir o espaço entre eles.

**5. Qual é a função das propriedades `padding` e `margin`? Como elas diferem?**

* **Padding** - Limpa uma área ao redor do conteúdo. O padding é transparente
* **Margem** - Limpa uma área fora da borda. A margem é transparente

**6. Como o posicionamento `relative` difere do `absolute` em CSS?**

Um elemento com position: relative;é posicionado em relação à sua posição normal.

Definir as propriedades top, right, bottom e left de um elemento relativamente posicionado fará com que ele seja ajustado para longe de sua posição normal. Outros conteúdos não serão ajustados para caber em nenhuma lacuna deixada pelo elemento.

Um elemento position: absolute;é posicionado em relação ao ancestral posicionado mais próximo (em vez de posicionado em relação à janela de visualização, como fixo).

Entretanto, se um elemento posicionado de forma absoluta não tiver ancestrais posicionados, ele usará o corpo do documento e se moverá junto com a rolagem da página.

**Observação:** elementos posicionados de forma absoluta são removidos do fluxo normal e podem sobrepor elementos.

**7. O que são Media Queries e como elas são utilizadas no design responsivo?**

Media query é uma técnica CSS introduzida no CSS3.

Ele usa a @media regra para incluir um bloco de propriedades CSS somente se uma determinada condição for verdadeira.

**8. Explique a diferença entre unidades relativas (`em`, `rem`, `%`) e unidades absolutas (`px`, `cm`).**

**Rem -** O rem é uma das unidades de medida mais usadas no desenvolvimento front-end, porém nem todos entendem como essa medida funciona. Então, vamos entendê-la, 1rem é igual ao tamanho padrão da fonte da raiz(root) do documento, normalmente 16px, logo 1rem é igual à 16px.

Se alterarmos a medida da fonte na raiz(root), alteramos o tamanho do rem, isso acontece com dispositivos de tamanho menores e maiores, pois 1px em um celular, por exemplo, é menor que 1px de uma televisão. Dessa forma nosso texto se torna mais adaptável a vários tamanhos de telas.

**Em**-Essa medida se comporta semelhante ao rem, a diferença dela é que 1em se baseia no tamanho do elemento pai de um parágrafo.

**Porcentagem-**Essa unidade de medida é um dos tipos mais utilizados, quando trabalhamos com CSS. Isso porque, ela se adapta aos diferentes tamanhos de tela. O uso dessa unidade é feito da seguinte forma: quando declaramos que um elemento terá 100% de largura, estamos dizendo que esse mesmo elemento, tem a largura total do elemento que o envolve. Vamos ver um exemplo para ficar mais claro.

**PX -** é uma unidade absoluta e é adequada quando você precisa definir tamanhos exatos.

**Centímetro**- é uma unidade de medida muito mais utilizada no mundo real do que no desenvolvimento front-end, o uso dessa medida é inviável para se definir o tamanho de um texto ou título, pois 1cm pode ser muito grande para a tela de um celular.

**9. Qual é a função da propriedade `display`? Compare as opções `block`, `inline` e `inline-block`.**

## Elementos inline- Um elemento embutido NÃO começa em uma nova linha e ocupa apenas a largura necessária.

Este é um elemento <span> embutido dentro de um parágrafo.

Exemplos de elementos inline:

* <período>
* <um>
* <img>  
  inline Exibe um elemento como um elemento embutido.

**Block** - Exibe um elemento como um elemento de bloco.

**inline-block** - Exibe um elemento como um contêiner de bloco de nível inline. O elemento em si é formatado como um elemento embutido, mas você pode aplicar valores de altura e largura.

**10. Como as pseudo-classes são usadas em CSS? Dê dois exemplos práticos.**

## O que são pseudoclasses?

Uma pseudoclasse é usada para definir um estado especial de um elemento.

Por exemplo, pode ser usado para:

* Estilizar um elemento quando um usuário passa o mouse sobre ele
* Estilize links visitados e não visitados de forma diferente
* Estilize um elemento quando ele receber foco.

a.highlight:hover {  
  color: #ff0000;} muda a seta para mãozinha.

a:hover {  
  color: #FF00FF;} muda a cor do link quando passa o mouse.

**11. Crie um estilo CSS que aplique a cor azul a todos os títulos `h1` de uma página.**

h1{color:blue;}

**12. Estilize um parágrafo `p` com uma margem de 20 pixels ao redor e um preenchimento de 10**

**pixels.**

P{padding:10px; margin:20px;}

**13. Utilize CSS para definir que todos os links `a` mudem a cor para verde quando o mouse passa**

**sobre eles.**

A:hover{color:green;}

**14. Defina um layout de página simples com três colunas usando `display: flex`.**

.esquerda { background-color: #f8b400; } .central { background-color: #34b7f1; color: #fff; } .direita { background-color: #32a852; color: #fff; }

 display: flex na container faz com que os filhos (as colunas) sejam dispostos em linha (horizontalmente) e de forma flexível.

 flex: 1 nas colunas faz com que cada coluna ocupe a mesma quantidade de espaço dentro do container.

 height: 100vh no container faz com que ele ocupe toda a altura da viewport.

 padding e box-sizing garantem que o conteúdo tenha algum espaçamento interno e que o padding não afete o tamanho total das colunas.

**15. Crie uma regra CSS que aplique uma borda de 2 pixels, sólida e preta, em todos os elementos**

**`div`.**

div{border:2px solid black;}

**16. Estilize uma classe chamada `.card` para ter uma sombra de caixa e bordas arredondadas.**

.card{border-radius:10px box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2);}

**17. Faça um menu de navegação horizontal, onde os itens estejam dispostos em linha usando**

**`display: inline-block`.**

.menu li { display: inline-block; /\* Faz com que os itens fiquem em linha \*/ }

<nav>

<ul class="menu">

<li><a href="#home">Home</a></li>

<li><a href="#about">Sobre</a></li>

<li><a href="#services">Serviços</a></li>

<li><a href="#contact">Contato</a></li>

</ul>

</nav>

**18. Utilize uma Media Query para alterar o fundo da página para cinza quando a largura da tela for**

**inferior a 600 pixels.**

body { background-color: white; /\* Cor de fundo padrão \*/ }

@media (max-width: 600px) { body { background-color: gray; /\* Cor de fundo alterada para cinza \*/ }

**19. Crie um estilo CSS para posicionar um rodapé (`footer`) fixo na parte inferior da página.**

footer { position: fixed; /\* Fixa o rodapé na posição especificada \*/ bottom: 0; /\* Alinha o rodapé na parte inferior da página \*/ left: 0; /\* Alinha o rodapé ao lado esquerdo da página \*/ width: 100%; /\* Faz o rodapé ocupar toda a largura da página \*/ background-color: #333; /\* Cor de fundo do rodapé (opcional) \*/ color: #fff; /\* Cor do texto no rodapé (opcional) \*/ text-align: center; /\* Centraliza o texto dentro do rodapé (opcional) \*/ padding: 10px; /\* Espaçamento interno do rodapé (opcional) \*/ box-sizing: border-box; /\* Inclui padding e border no cálculo da largura \*/ }

body { margin: 0; padding-bottom: 50px; /\* Adiciona um padding no fundo do body para evitar sobreposição \*/ }

**20. Escreva um código CSS que centralize um elemento `div` dentro de seu contêiner pai tanto**

**horizontalmente quanto verticalmente.**

<!DOCTYPE html>

<html lang="pt-BR">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Centralizar com Flexbox</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="container">

<div class="centered-div">Conteúdo Centralizado</div>

</div>

</body>

</html>

/\* Estilo para o contêiner pai \*/

.container {

display: flex; /\* Define o contêiner como um flex container \*/

justify-content: center; /\* Centraliza o conteúdo horizontalmente \*/

align-items: center; /\* Centraliza o conteúdo verticalmente \*/

height: 100vh; /\* Define a altura do contêiner (opcional, mas útil para visualização) \*/

}

/\* Estilo para a div a ser centralizada \*/

.centered-div {

width: 200px; /\* Largura da div (ajuste conforme necessário) \*/

height: 100px; /\* Altura da div (ajuste conforme necessário) \*/

background-color: #f39c12; /\* Cor de fundo da div (opcional) \*/

}