

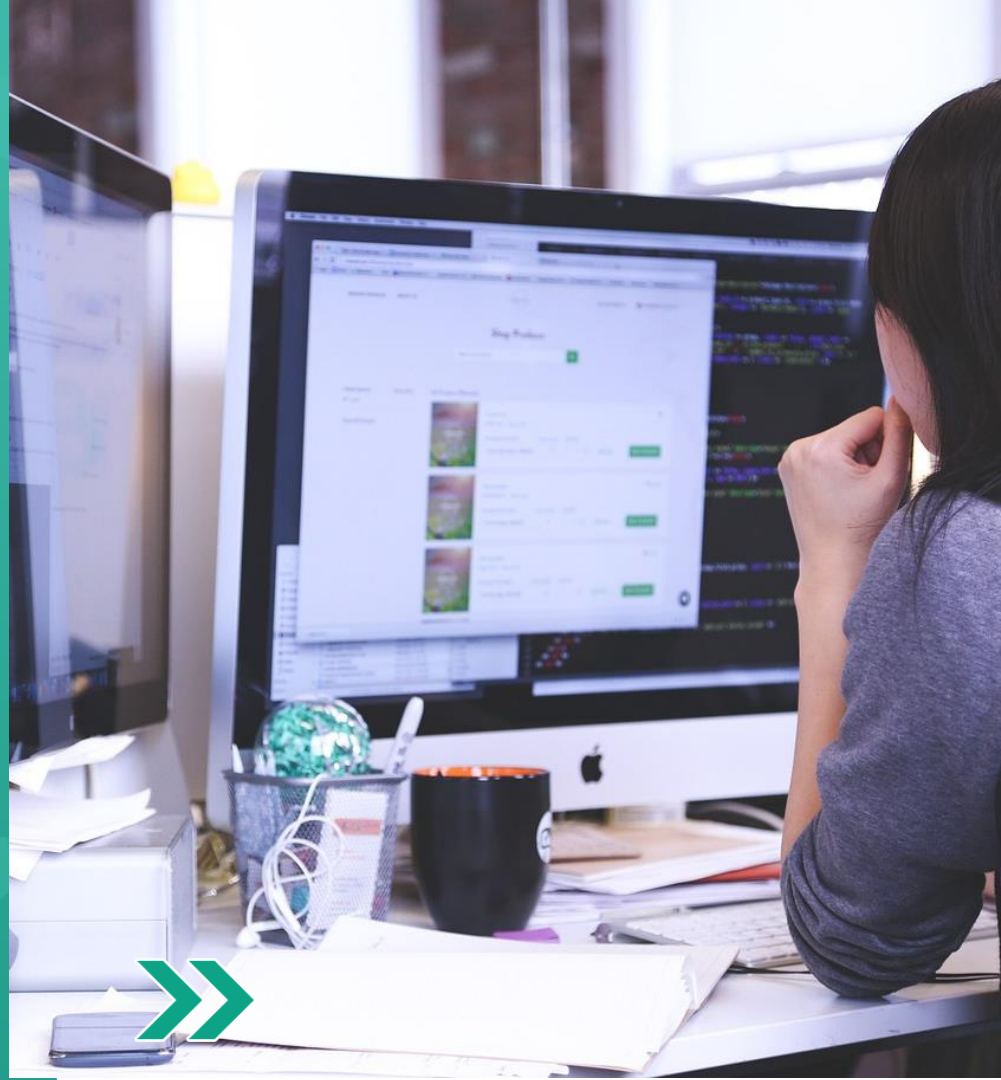


escola  
britânica de  
artes criativas  
& tecnologia

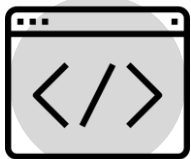
# Profissão: Engenheiro Front-End



# BOAS PRÁTICAS



# Boas práticas de CSS



Confira boas práticas da comunidade de Front-End por assunto relacionado às aulas.

- **Determine regras de CSS**
- **Aplique a metodologia SMACSS**
- **Aplique a metodologia BEM**



# Determine regras de CSS

As **pseudoclasses** oferecem uma maneira poderosa de estilizar elementos com base em diferentes estados e características, permitindo uma personalização mais avançada da aparência dos elementos HTML. Recomendamos sempre consultar a documentação do CSS para ver a lista completa de pseudoclasses disponíveis e aprender como usá-las corretamente. Acompanhe alguns exemplos.



## Pseudoclasses:

- **:hover**: Aplica estilos quando o elemento é "hovered" (quando o cursor do mouse está sobre ele).
- **:active**: Aplica estilos quando o elemento está ativo (por exemplo, quando um botão é clicado e mantido pressionado).
- **:focus**: Aplica estilos quando o elemento está em foco (por exemplo, quando um campo de formulário é selecionado).
- **:first-child**: Aplica estilos ao primeiro elemento filho de seu elemento pai.
- **:last-child**: Aplica estilos ao último elemento filho de seu elemento pai.
- **:nth-child()**: Permite selecionar um elemento específico com base em sua posição em relação aos elementos irmãos. Por exemplo, `:nth-child(2)` seleciona o segundo elemento filho.
- **:nth-of-type()**: Permite selecionar um elemento específico com base em sua posição em relação aos elementos irmãos do mesmo tipo. Por exemplo, `:nth-of-type(odd)` seleciona os elementos ímpares.

# Determine regras de CSS

Os **pseudo elementos** têm suas próprias propriedades de estilo específicas que podem ser aplicadas. É importante consultar a documentação do CSS para ver a lista completa de pseudoelementos disponíveis e entender como usá-los corretamente. Acompanhe alguns exemplos.



## Pseudo elementos:

- **::before**: Permite adicionar conteúdo antes do conteúdo do elemento selecionado. Pode ser usado para adicionar ícones, elementos decorativos ou texto adicional.
- **::after**: Permite adicionar conteúdo após o conteúdo do elemento selecionado. Assim como ::before, pode ser usado para adicionar elementos ou texto adicionais.
- **::first-line**: Aplica estilos somente à primeira linha de texto dentro do elemento selecionado. Isso permite a formatação específica da primeira linha de um parágrafo, por exemplo.
- **::first-letter**: Aplica estilos somente à primeira letra do conteúdo dentro do elemento selecionado. Isso pode ser usado para criar efeitos de drop cap (letra capital ampliada) ou estilizar a primeira letra de um parágrafo de forma especial.
- **::selection**: Permite estilizar a seleção de texto dentro do elemento. É possível definir cores de fundo e texto diferentes para a seleção.

# Aplique a metodologia SMACSS

Acompanhe algumas boas práticas relacionadas ao SMACSS que podem ajudar a criar um código CSS organizado e de fácil manutenção.



- **Separação de responsabilidades:**  
Divida o código CSS em diferentes módulos e componentes, cada um sendo responsável por uma área específica da interface. Isso facilita a compreensão do código e permite a reutilização em diferentes partes do site.
- **Estrutura de arquivos:**  
Organize seus arquivos CSS em uma estrutura lógica. Geralmente, é útil ter uma pasta para os estilos base (base/), outra para os estilos de layout (layout/), uma para os módulos (modules/), entre outras que possam ser relevantes para o seu projeto.
- **Nomeação de classes:**  
Use nomes de classes descritivos e significativos para seus elementos HTML. Opte por nomes que indiquem o propósito ou a função do elemento, em vez de se basear apenas em sua aparência visual.

# Aplique a metodologia SMACSS

Acompanhe algumas boas práticas relacionadas ao SMACSS que podem ajudar a criar um código CSS organizado e de fácil manutenção.



- **Modularidade:**  
Crie módulos independentes que possam ser facilmente reutilizados. Isso significa que os estilos de um módulo não devem depender dos estilos de outros módulos. Dessa forma, você pode movê-los ou modificá-los sem quebrar o layout.
- **Evite estilos diretos em elementos HTML:**  
Em vez de aplicar estilos diretamente em elementos HTML usando o atributo "style", utilize classes CSS para estilizar os elementos. Isso mantém o CSS separado do HTML e facilita a manutenção.
- **Cascata e especificidade:**  
Evite o uso excessivo de especificidade em seus seletores CSS. Isso pode tornar o código mais difícil de entender e alterar posteriormente. Procure utilizar seletores simples e adicionar classes específicas apenas quando necessário.

# Aplique a metodologia SMACSS

Acompanhe algumas boas práticas relacionadas ao SMACSS que podem ajudar a criar um código CSS organizado e de fácil manutenção.



- **Comentários:**  
Inclua comentários no código CSS para explicar o propósito ou a função de certos estilos. Isso facilita a compreensão do código por outros desenvolvedores e por você mesmo no futuro.
- **Minificação e concatenação:**  
Para melhorar o desempenho do seu site, considere minificar e concatenar os arquivos CSS antes de implantar em produção. Isso reduz o tamanho dos arquivos e melhora o tempo de carregamento da página.



# Aplique a metodologia SMACSS



## **ParentNode:**

A propriedade parentNode retorna null para o nó raiz do documento (o elemento <html>), pois ele não tem um elemento pai. Além disso, se um elemento for removido do documento, seu parentNode também se tornará null. Portanto, é uma boa prática verificar se o parentNode é diferente de null antes de interagir com ele.



# Aplique a metodologia BEM

Acompanhe algumas boas práticas introdutórias para o uso do BEM.



- **Nomes descritivos:**  
Use nomes de classes descritivos e significativos que representem claramente a função e o propósito do elemento ou componente. Evite nomes genéricos ou ambíguos.
- **Separação por hífen:**  
Utilize hífen para separar os diferentes componentes e modificadores em uma classe BEM. Por exemplo, `block__element--modifier`.
- **Estruturação modular:**  
Divida o código CSS em módulos independentes e reutilizáveis. Cada módulo deve corresponder a um bloco BEM. Isso facilita a manutenção e o entendimento do código.
- **Evite estilização direta de elementos:**  
Evite estilizar diretamente elementos HTML usando seletores de tag. Em vez disso, aplique estilos usando classes BEM para garantir a separação de preocupações e facilitar a reutilização de estilos.

# Aplique a metodologia BEM

Acompanhe algumas boas práticas introdutórias para o uso do BEM.



- **Evite a cascata excessiva:**  
Mantenha as regras CSS específicas para cada bloco ou elemento, evitando a aplicação de estilos que afetem outros componentes ou elementos não relacionados.
- **Use modificadores para variações:**  
Utilize os modificadores BEM para criar variações ou estados específicos de um componente. Isso permite alterar a aparência ou comportamento sem a necessidade de duplicar código.
- **Aninhe elementos BEM adequadamente:**  
Se um elemento BEM estiver contido em outro elemento BEM, ele deve ser aninhado corretamente em sua estrutura HTML correspondente. Isso reflete a hierarquia e relacionamento dos componentes.
- **Evite a sobrequalificação dos seletores:**  
Evite o uso de seletores muito específicos e complexos que tornem o código difícil de entender ou de modificar. Mantenha os seletores o mais simples e diretos possível.

# Bons estudos!

