JavaScript: Fundamentos para Desenvolvimento Web Interativo - Turma 2025B

1.3 Estratégias para carregamento eficiente de códigos JavaScript

Carregar scripts JavaScript na ordem correta e no momento apropriado é crucial, mas este processo pode ser desafiador. Scripts que manipulam o Document Object Model (DOM) podem encontrar problemas se forem carregados e executados antes que os elementos HTML correspondentes estejam disponíveis no DOM. Isso ocorre porque, por padrão, o HTML é processado na ordem em que aparece no documento, e se um script tentar acessar elementos que ainda não foram carregados, isso resultará em erros.

Por exemplo, quando o JavaScript é inserido no cabeçalho de uma página (**<head>**), ele será carregado antes do conteúdo do corpo (**<body>**), o que pode causar erros se o script tentar interagir com elementos que ainda não foram renderizados.

Veremos abaixo algumas soluções para o carregamento sequencial de scripts JavaScript em páginas web.

Uso do atributo defer

O atributo **defer** é uma solução eficaz para scripts que não dependem de outros scripts e que necessitam acessar o DOM. Ao adicionar **defer** à tag **<script>**, o navegador irá continuar a processar o HTML, carregando o script em paralelo, mas só o executará após todo o documento ter sido carregado.

Veja sua sintaxe básica no quadro abaixo:

```
<script src="script.js" defer></script>
```

Uso do atributo async

Diferente de **defer**, o atributo **async** também permite que o script seja baixado em paralelo com a renderização da página, mas ele será executado assim que estiver disponível, o que pode ser antes ou depois que o HTML estiver completamente carregado. Isso é ideal para scripts que não interagem com o DOM ou que não dependem da ordem completa de carregamento da página.

Veja sua sintaxe básica no quadro abaixo:

```
<script src="script.js" async></script>
```

Carregar scripts no final do corpo do documento HTML

Colocar a tag **<script>** no final do corpo do documento, imediatamente antes do fechamento da tag **</body>**, é uma prática comum para garantir que todos os elementos HTML já estejam carregados antes da execução do script.

Veja um exemplo no quadro abaixo:

```
<body>
  <!-- Conteúdo da página -->
  <script src="meu-script.js"></script>
</body>
```

OBS: Quando você usa scripts externos, pode usar os atributos **async** ou **defer** para controlar como e quando o JavaScript é carregado, melhorando ainda mais o desempenho da página.

Event listener para DOMContentLoaded

Uma técnica avançada envolve adicionar um ouvinte de evento (**event listener**) no JavaScript para detectar quando o DOM está totalmente carregado, o que é útil para scripts internos. Um '**event listener**' é usado para monitorar eventos específicos no navegador. Neste caso, o evento em questão é o '**DOMContentLoaded**', que é disparado pelo navegador assim que o conteúdo do

DOM (Document Object Model) está completamente carregado, indicando que todos os elementos do HTML estão prontos para serem manipulados.

O JavaScript contido dentro do bloco de código associado a este ouvinte de eventos não será executado até que o evento '**DOMContentLoaded**' seja efetivamente disparado. Isso assegura que qualquer manipulação do DOM realizada pelo JavaScript só ocorra quando for seguro fazê-lo, evitando erros de tentativas de acesso a elementos que ainda não foram completamente carregados na página.

Veja um exemplo no quadro abaixo:

```
document.addEventListener('DOMContentLoaded', function () {
    // Seu código aqui que manipula o DOM
});
```

◄ 1.2 Integração do JavaScript com o HTML

Seguir para...

1.4 Teste seus conhecimentos ►