PADO

Labs

INTRODUÇÃO JAVASCRIPT

Desenvolvimento Web - Aula VI

Contato

Professor


David Krepsky


david@hausenn.com.br


DKrepsky

Monitor


Kevin Araujo


kevin.araujo@hausenn.tech


NivekDesign

Java Script

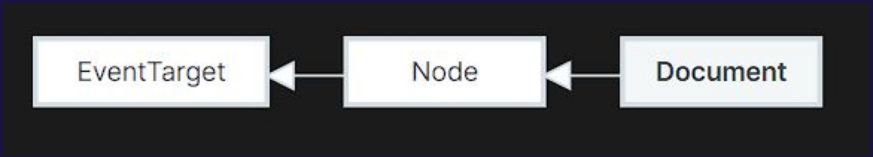
- A classe Document
- Criação de elementos dinâmicos
- Árvore de nós e elementos
- Adição e remoção de elementos a árvore
- Manipulação de atributos
- A classe Window
- Evento onload
- Bonus: Web Apis

A Classe Document

A Document interface representa qualquer página da web carregada no navegador e serve como ponto de entrada para o conteúdo da página da web, que é a árvore DOM .

A árvore DOM inclui elementos como <body> e <table>, entre muitos outros. Ele fornece funcionalidade global ao documento, como obter a URL da página e criar novos elementos no documento.

A Classe Document



A Document interface descreve as propriedades e métodos comuns para qualquer tipo de documento. Dependendo do tipo do documento (por exemplo , HTML , XML , SVG, …), uma API maior está disponível: documentos HTML, servidos com o "text/html" tipo de conteúdo, também implementam a HTMLDocument interface, enquanto documentos XML e SVG implementam a XMLDocument interface.

A Classe Document

Propriedades

Document.all

Dá acesso a todos os elementos do documento. É uma interface legada não padrão, você deve usar o método Document.getElementById()alternativo.

Document.async

Usado como document.load que indica uma requisição assíncrona.

Criação de elementos dinâmicos

Em um documento HTML, o método Document.createElement() cria o elemento HTML especificado ou um HTMLUnknownElement se o nome do elemento dado não for conhecido.

Sintaxe:

```
var elemento = document.createElement(nomeDaTag);
```

elemento é o objeto Element criado.

nomeDaTag é uma string que especifica o tipo do elemento a ser criado.

Criação de elementos dinâmicos

Examples

Crie um elemento <p> e anexe-o ao documento:

```
const para = document.createElement("p");
para.innerText = "This is a paragraph";
document.body.appendChild(para);
```

Crie um elemento <p> e anexe-o a um elemento

```
const para = document.createElement("p");
para.innerHTML = "This is a paragraph.";
document.getElementById("myDIV").appendChild(para);
```

Árvore de Nós e Elementos

https://www.w3schools.com/jsref/prop_element_children.asp

Nota:

- Diferença entre nós e elementos
- Mostrar estrutura de árvore dos nós
- Acesso aos nós com *children, parentElement, firstElementChild, lastElementChild parentNode, last*

Árvore de Nós e Elementos

No HTML DOM (Document Object Model), um documento HTML é uma coleção de nós com (ou sem) nós filhos. Nós são nós de elemento, nós de texto e nós de comentário. Os espaços em branco entre os elementos também são nós de texto. Elementos são apenas nós de elementos.

Adição e remoção de elementos a árvore

Node.appendChild

Adiciona um nó ao final da lista de filhos de um nó pai especificado. Se o nó já existir no documento, ele é removido de seu nó pai atual antes de ser adicionado ao novo pai.

Sintaxe:

```
var filho = elemento.appendChild(filho);
```

appendChild()

O método appendChild devolve uma referência ao nó adicionado.

```
// Cria um novo elemento de parágrafo e adiciona-o ao final do documento
var p = document.createElement("p");
document.body.appendChild(p);
```

Se filho é uma referência a um nó existente no documento, appendChild vai movê-lo de sua posição atual para a nova posição (i.e, não é necessário remover o nó de seu pai atual antes de adicioná-lo a outro nó).

contains()

Indica se um nó é um descendente de um dado nó.

Sintaxe:

```
node.contains( otherNode )
```

node é o nó que está sendo comparado.

otherNode é o nó contra o qual está sendo comparado.

O valor de retorno é true se otherNode é um descendente de um nó ou o próprio nó. Caso contrário o valor de retorno é false.

insertBefore()

O método Node.insertBefore() insere um nó antes do nó de referência como um filho de um nó pai especificado. Se o filho especificado for uma referência a um nó existente no documento, insertBefore() o moverá de sua posição atual para a nova posição (não há necessidade de remover o nó de seu nó pai antes de anexá-lo a outro nó).

Sintaxe:

```
var elementoInserido = elementoPai.insertBefore(novoElemento,
elementoDeReferencia);
```

insertBefore()

Sintaxe:

```
var elementoInserido = elementoPai.insertBefore(novoElemento,
elementoDeReferencia);
```

- elementoInserido O nó sendo inserido, que é novoElemento.
- elementoPai Pai do nó recentemente inserido.
- novoElemento O nó a ser inserido.
- elementoDeReferencia O nó antes do qual o novoElemento será inserido.

removeChild()

Remove um nó filho do DOM. Devolve o nó removido.

Sintaxe:

```
var filhoRemovido = elemento.removeChild(filho);
elemento.removeChild(filho);
```

- filho é o nó filho a ser removido do DOM.
- elemento é o nó pai de filho.
- filhoRemovido contém uma referência ao nó filho removido. filhoRemovido === filho.

replaceChild()

Substitui o elemento filho especificado por outro.

Sintaxe:

```
replacedNode = parentNode.replaceChild(newChild, oldChild);
```

- newChild é o novo elemento que será inserido no lugar do oldChild. Se já existir no DOM, será removido primeiro para depois ser inserido.
- oldChild é o elemento existente que será substituído.
- replacedNode é elemento substituído. É o mesmo elemento que oldChild.

Manipulação de atributos

setAttribute()

Adiciona um novo atributo ou modifica o valor de um atributo existente num elemento específico.

Sintaxe:

```
element.setAttribute(name, value);
```

- name é o nome do atributo como string.
- value é o novo valor desejado do atributo

setAttribute()

Exemplo

setAttribute() é usado para definir o atributo disabled em <button>, desabilitado-o.

```
<button>Hello World</button>
```

```
var b = document.querySelector("button");

b.setAttribute("disabled", "disabled");
```

getAttribute()

getAttribute() retorna o valor de um argumento específico do elemento. Se o atributo não existir, o valor retornado será null ou "" (string vazia).

Sintaxe:

```
var atributo = element.getAttribute(nomeDoAtributo);
```

- atributo é uma string contendo o valor do nomeDoAtributo.
- nomeDoAtributo é o nome do atributo cujo deseja se obter o valor.

hasAttribute()

O Element.hasAttribute() método retorna um valor booleano indicando se o elemento especificado possui o atributo especificado ou não.

Sintaxe:

```
hasAttribute(name)
```

Parâmetro: é uma string que representa o nome do atributo.

Valor de retorno: Um Booleano,

removeAttribute()

removeAttributeremova um atributo de um elemento específico.

Sintaxe:

```
element.removeAttribute(attrName);
```

attrName é o nome, em formato de texto (string), do atributo a ser removido do element.

removeAttribute()

Exemplo:

```
// <div id="div1" align="left" width="200px">
document.getElementById("div1").removeAttribute("align");
// agora: <div id="div1" width="200px">
```

Você deve usar removeAttribute ao invés de atribuir nullao atributo setAttribute . Tentar removedor um atributo que não existe no elemento não fará que uma exceção seja lançada.

A classe Window

O objeto window representa uma janela que contém um elemento DOM; a propriedade document aponta para o documento DOM document carregado naquela janela. Uma janela para um dado documento pode ser obtido usando a propriedade document.defaultView.

Esta seção provê uma breve referência a todos os métodos, propriedades e eventos disponíveis através do objeto DOM window. O objeto window implementa a interface Window, o qual herda da interface AbstractView

A classe Window

Evento onload

O evento de load é acionado quando um recurso e seus recursos dependentes terminaram de carregar.

Exemplo:

```
<script>
  window.addEventListener("load", function(event) {
    console.log("Todos os recursos terminaram o carregamento!");
  });
</script>
```

Bluetooth API

A API Web Bluetooth oferece a capacidade de conectar e interagir com periféricos Bluetooth Low Energy.

- Bluetooth: Retorna a Promise para um BluetoothDeviceobjeto com as opções especificadas.
- BluetoothCharacteristicProperties: Fornece propriedades de um determinado BluetoothRemoteGATTCharacteristic.

Bluetooth API

- BluetoothDevice: Representa um dispositivo Bluetooth dentro de um ambiente de execução de script específico.
- BluetoothRemoteGATTCharacteristic: Representa uma Característica GATT, que é um elemento de dados básico que fornece mais informações sobre o serviço de um periférico.
- BluetoothRemoteGATTDescriptor: Representa um Descritor GATT, que fornece mais informações sobre o valor de uma característica.

Bluetooth API

- BluetoothRemoteGATTServer: Representa um GATT Server em um dispositivo remoto.
- BluetoothRemoteGATTService: Representa um serviço fornecido por um servidor GATT, incluindo um dispositivo, uma lista de serviços referenciados e uma lista das características desse serviço.

Canvas API

API do Canvas prova maneiras de desenhar gráficos via JavaScript e via elemento HTML <canvas> . Entre outras, ele pode ser utilizado para animação, gráficos de jogos, visualização de dados, manipulação de fotos e processamento de vídeo em tempo real.

A Canvas API focando amplamente em gráficos 2D. A API WebGL , que também usa o elemento <canvas>, desenha gráficos 2D e 3D acelerados por hardware.

Clipboard API

A API da área de transferência fornece a capacidade de responder a comandos da área de transferência (cortar, copiar e colar), bem como ler e gravar de forma assíncrona na área de transferência do sistema.

- Clipboard: Fornece uma interface para leitura e gravação de texto e dados de ou para a área de transferência do sistema. A especificação refere-se a isso como 'API da área de transferência assíncrona'.

Clipboard API

- ClipboardEvent: Representa eventos que fornecem informações relacionadas à modificação da área de transferência, ou seja cut, copy, e pasteeventos. A especificação refere-se a isso como a 'API de evento da área de transferência'.
- ClipboardItem: Representa um formato de item único, usado ao ler ou gravar dados.

File API

A API de arquivo permite que aplicativos da Web acessem arquivos e seus conteúdos.

Os aplicativos da Web podem acessar os arquivos quando o usuário os disponibiliza, seja usando um elemento de arquivo<input> ou arrastando e soltando .

Conjuntos de arquivos disponibilizados dessa maneira são representados como FileListobjetos, que permitem que um aplicativo da Web recupere Fileobjetos individuais. Por sua vez File, os objetos fornecem acesso a metadados, como nome, tamanho, tipo e data da última modificação do arquivo.

File API

Interfaces

- Blob: Representa um "Binary Large Object", ou seja, um objeto semelhante a um arquivo de dados brutos imutáveis; a Blobpode ser lido como texto ou dados binários, ou convertido em a ReadableStreampara que seus métodos possam ser usados para processar os dados.
- File: Fornece informações sobre um arquivo e permite que o JavaScript em uma página da Web acesse seu conteúdo.

File API

- FileList: Retornado pela filespropriedade do <input>elemento HTML; isso permite acessar a lista de arquivos selecionados com o <input type="file">elemento. Também é usado para uma lista de arquivos inseridos no conteúdo da Web ao usar a API de arrastar e soltar; consulte o DataTransferobjeto para obter detalhes sobre esse uso.
- FileReader: Permite que os aplicativos da Web leiam de forma assíncrona o conteúdo dos arquivos (ou buffers de dados brutos) armazenados no computador do usuário, usando objetos Fileou Blobpara especificar o arquivo ou os dados a serem lidos.

File API

- FileReaderSync: Permite que os aplicativos da Web leiam de forma síncrona o conteúdo dos arquivos (ou buffers de dados brutos) armazenados no computador do usuário, usando objetos Fileou Blobpara especificar o arquivo ou os dados a serem lidos.

IndexedDB

A API de acesso ao sistema de arquivos permite recursos de leitura, gravação e gerenciamento de arquivos.

Essa API permite a interação com arquivos no dispositivo local de um usuário ou em um sistema de arquivos de rede acessível ao usuário. A funcionalidade principal desta API inclui leitura de arquivos, gravação ou salvamento de arquivos e acesso à estrutura de diretórios.

Storage

A localStoragepropriedade read-only da windowinterface permite acessar um Storageobjeto para a Documentorigem do ; os dados armazenados são salvos nas sessões do navegador. localStorageé semelhante a sessionStorage, exceto que, embora localStorageos dados não tenham tempo de expiração, sessionStorageos dados são limpos quando a sessão da página termina — ou seja, quando a página é fechada. (localStorageos dados de um documento carregado em uma sessão de "navegação privada" ou "anônima" são apagados quando a última guia "privada" é fechada.)

Sensor API

As APIs do Sensor são um conjunto de interfaces construídas para um design comum que expõem os sensores do dispositivo de forma consistente à plataforma web.

Embora a especificação da API de sensor genérico defina uma Sensorinterface, como desenvolvedor da Web, você nunca a usará. Em vez disso, você usará uma de suas subclasses para recuperar tipos específicos de dados do sensor. Por exemplo, a accelerometerinterface retorna a aceleração do dispositivo ao longo dos três eixos no momento da leitura.

Web Crypto API

A Web Crypto API é uma interface que permite que um script use primitivas criptográficas para construir sistemas usando criptografia.

Alguns navegadores implementaram uma interface chamada Cryptosem tê-la bem definida ou ser criptograficamente sólida. Para evitar confusão, os métodos e propriedades desta interface foram removidos dos navegadores que implementam a API Web Crypto, e todos os métodos da API Web Crypto estão disponíveis em uma nova interface: SubtleCrypto. A Crypto.subtlepropriedade dá acesso a um objeto que a implementa.

Websockets API

A API WebSocket é uma tecnologia avançada que possibilita a abertura de uma sessão de comunicação interativa bidirecional entre o navegador do usuário e um servidor. Com essa API, você pode enviar mensagens para um servidor e receber respostas orientadas a eventos sem precisar pesquisar uma resposta no servidor.

Bonus: Web Apis

A API Web Serial fornece uma maneira para sites lerem e gravarem em dispositivos seriais. Esses dispositivos podem ser conectados através de uma porta serial, ou ser dispositivos USB ou Bluetooth que emulam uma porta serial.

A API Web Serial faz parte de um conjunto de APIs que permitem que sites se comuniquem com periféricos conectados ao computador de um usuário. Ele fornece a capacidade de se conectar a dispositivos que são exigidos pelo sistema operacional para se comunicar por meio da API serial, em vez de USB, que pode ser acessado por meio do WebUSB API, ou dispositivos de entrada que podem ser acessados pelo WebHID API.

Bonus: Web Apis

IndexedDB é uma API de baixo nível para armazenamento do lado do cliente de quantidades significativas de dados estruturados, incluindo arquivos/blobs. Essa API usa índices para permitir pesquisas de alto desempenho desses dados. Embora o Web Storage seja útil para armazenar quantidades menores de dados, é menos útil para armazenar quantidades maiores de dados estruturados. IndexedDB fornece uma solução. Esta é a página principal da cobertura do IndexedDB do MDN — aqui fornecemos links para a referência completa da API e guias de uso, detalhes de suporte do navegador e algumas explicações dos principais conceitos.

Geolocation API

A API de geolocalização permite que o usuário forneça sua localização para aplicativos da web, se assim o desejar. Por motivos de privacidade, é solicitada permissão ao usuário para relatar informações de localização. WebExtensions que desejam usar o Geolocationobjeto devem adicionar a "geolocation"permissão ao seu manifesto. O sistema operacional do usuário solicitará que o usuário permita o acesso ao local na primeira vez que for solicitado.

Web Serial

A API de entradas de arquivos e diretórios simula um sistema de arquivos local no qual aplicativos da Web podem navegar e acessar arquivos. Você pode desenvolver aplicativos que lêem, gravam e criam arquivos e/ou diretórios em um sistema de arquivos virtual em área restrita.

Exercício 1 - Aplicação TODO List

Concluir a aplicação TODO

Referências

1. https://www.caelum.com.br/apostila/apostila-html-css-javascript.pdf
2. https://www.scriptbrasil.com.br/download/apostila/837/
3. https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch#uploading_json_data
4. https://www.w3schools.com/js
5. https://developer.mozilla.org/en-US/docs/Web/API
6.

PADO

Labs