Documentação do arquivo viewVehicles.js

Introdução

O arquivo `viewVehicles.js` é um componente Lightning Web Component (LWC) desenvolvido para exibir uma lista de veículos disponíveis e permitir que os usuários realizem a compra de um veículo. Ele utiliza métodos Apex para buscar os dados dos veículos e persistir a compra no Salesforce.

Descrição

Este arquivo contém a lógica para:

1. Buscar a lista de veículos disponíveis através de um método Apex.
2. Exibir os veículos na interface do usuário.
3. Permitir que o usuário selecione e compre um veículo.
4. Exibir notificações (toasts) para informar o sucesso ou falha das operações.

Estrutura

O arquivo é estruturado como um componente LWC, contendo:

- Importações de módulos e métodos necessários.
- Propriedades e variáveis para armazenar os dados dos veículos.
- Métodos para buscar os veículos, realizar a compra e exibir notificações.

Dependências

O arquivo depende dos seguintes recursos:

- `@salesforce/apex/Vehicles.getVehicles`: Método Apex para buscar os veículos.
- `@salesforce/apex/Vehicles.persistPurchase`: Método Apex para persistir a compra de um veículo.
- `lightning/platformShowToastEvent`: Módulo para exibir notificações na interface do usuário.

Imports

Os seguintes módulos e métodos são importados:

```
import { LightningElement, wire } from 'lwc';
import getVehicles from '@salesforce/apex/Vehicles.getVehicles';
import persistPurchase from '@salesforce/apex/Vehicles.persistPurchase';
import { ShowToastEvent } from "lightning/platformShowToastEvent";
```

Variáveis

- `vehicles`: Armazena a lista de veículos retornada pelo método Apex `getVehicles`.

Métodos

wiredVehicles

Este método é decorado com `@wire` e é responsável por buscar os veículos disponíveis. Ele utiliza o método Apex `getVehicles` e atualiza a variável `vehicles` com os dados retornados. Em caso de erro, o erro é registrado no console.

Parâmetros:

- `data`: Dados retornados pelo método Apex.

- `error`: Erro retornado pelo método Apex.

purchase

Este método é chamado quando o usuário tenta comprar um veículo. Ele utiliza o método Apex `persistPurchase` para persistir a compra. Em caso de sucesso, exibe uma notificação de sucesso. Em caso de erro, exibe uma notificação de erro.

Parâmetros:

- `event`: Evento disparado pelo clique do usuário, contendo o `id` do veículo selecionado.

showToast

Este método exibe uma notificação (toast) na interface do usuário.

Parâmetros:

- `title`: Título da notificação.
- `message`: Mensagem da notificação.
- `variant`: Tipo da notificação (`SUCCESS`, `ERROR`, etc.).

Exemplo

Abaixo está um exemplo de como o componente pode ser utilizado em um arquivo HTML:

```
<template>
    <lightning-card title="Lista de Veículos">
        <template if:true={vehicles}>
            <ul>
                <template for:each={vehicles} for:item="vehicle">
                    <li key={vehicle.Id}>
                        {vehicle.Name}
                        <button data-id={vehicle.Id} onclick={purchase}>Comprar</button>
                    </li>
                </template>
            </ul>
        </template>
        <template if:false={vehicles}>
            <p>Carregando veículos...</p>
        </template>
    </lightning-card>
</template>
```

Diagrama de Dependências

Abaixo está um diagrama de dependências do código:

```
classDiagram
    class ViewVehicles {
        +vehicles
        +wiredVehicles(data, error)
        +purchase(event)
        +showToast(title, message, variant)
    }
    class ApexMethods {
        +getVehicles()
        +persistPurchase(carId)
    }
    class ShowToastEvent {
        +title
        +message
        +variant
    }

    ViewVehicles --> ApexMethods : Utiliza
    ViewVehicles --> ShowToastEvent : Utiliza
```

Notas

- O método `wiredVehicles` utiliza a funcionalidade de `@wire` para buscar os dados de forma reativa.
- O método `purchase` utiliza o `dataset.id` para identificar o veículo selecionado.
- A função `showToast` facilita a exibição de notificações na interface do usuário.

Vulnerabilidades

- **Erro de conexão com Apex:** Caso o método Apex `getVehicles` ou `persistPurchase` falhe, o erro será registrado no console, mas não há tratamento adicional para lidar com falhas críticas.
- **Validação de `targetId`:** O método `purchase` depende do `dataset.id` para identificar o veículo. Caso o `id` não seja fornecido ou seja inválido, a operação falhará.
- **Recarga da página:** O código contém um comentário para recarregar a página (`location.reload()`), mas isso pode não ser ideal em termos de experiência do usuário.