Webdev - Módulo 2
Roteiro de aula

Aula 9 - Hard: Vendo o mapa

Tópicos da aula:

- Hoisting
- Closures
- Paradigma Funcional
- Array map
- View
- MVC

Objetivos da aula:

1. Compreender o que é a técnica do hoisting
2. Conceituar e utilizar closures
3. Entender o que é paradigma funcional em programação
4. Implementar array map
5. Entender o que são views e implementar em aplicação web
6. Identificar a lógica e a separação das responsabilidades na arquitetura MVC

Atividades da Aula

➔ **Atividade mão da massa: Mapeando**
- ◆ Acesse o link do Replit para fazer a atividade: https://replit.com/@luizgribeiro/atividademapeando#index.js
- ◆ Utilizando o método map no array "users" fornecido, crie novos arrays com as seguintes informações:
  - array nomes: apenas o nome das pessoa usuárias;
  - array nomeAbrev: apenas as abreviações (primeiras letras) do nome e sobrenome separados por ".";
  - amantesGatos: nomes abreviados das pessoas que tem gatos;
  - maiores: apenas pessoas maiores de 18 anos;
- ◆ Caso necessário use a documentação:

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Array/map

Momento Aprendizagem em Pares

- ➔ Esse momento é dedicado para vocês desenvolverem suas demandas e entregas para o curso.
- ➔ Utilize esse tempo da maneira que preferir, mas atente-se às aulas que você deve realizar as entregas.
  - ◆ **Dica:** Nas Propostas dos projetos, vocês encontram uma sugestão de organização para a realização das atividades.
  - ◆ **Lembre-se:** O momento de Aprendizagem em Pares é justamente para fazer trocas e aprender em comunidade, aproveite seus colegas e se desenvolvam juntos!
- ➔ **Entregas:**
  - ◆ Projeto individual: aula 5 - TECH
  - ◆ Projeto em grupo: aula 11 - TECH
  - ◆ Apresentação do projeto: aula 11 - TECH

Revisão da aula

- ➔ O **Closure** é a combinação de uma função agrupada (incluída) com referências ao seu estado circundante (o ambiente léxico). Em outras palavras, uma closure **fornece acesso ao escopo de uma função externa a partir de uma função interna.** Em JavaScript, os encerramentos são criados toda vez que uma função é criada, no momento da criação da função.

- ➔ Em JavaScript, **hoisting** permite que você use funções e variáveis antes de serem declaradas. Em outras palavras hoisting é um **mecanismo JavaScript em que variáveis e declarações de função são movidas para o topo de seu escopo antes da execução do código**. Inevitavelmente, isso significa que não importa onde as funções e variáveis sejam declaradas, elas são movidas para o topo de seu escopo, independentemente de seu escopo ser global ou local.

- ➔ **A programação funcional** é um estilo de programação em que as soluções são **funções simples e isoladas,** sem efeitos colaterais fora do escopo da função: ENTRADA => PROCESSO => SAÍDA

  A programação funcional é sobre:

1. **Funções isoladas** - não há dependência do estado do programa, que inclui variáveis globais que estão sujeitas a alterações
2. **Funções puras** - a mesma entrada sempre dá a mesma saída
3. **Funções com efeitos colaterais limitados** - quaisquer alterações ou mutações no estado do programa fora da função são cuidadosamente controladas

➔ Em JavaScript, um método integrado denominado **array.map()** é usado para criar um novo array modificado. Para fazer isso, ele percorre um array e invoca uma função para cada elemento do array. É muito útil em um cenário onde temos que implementar alguns procedimentos/ações em cada elemento do array. Por exemplo: multiplicar cada elemento da matriz por algum número específico ou encontrar a raiz quadrada de cada elemento da matriz e assim por diante.

➔ A **View** vai ser a **camada que vai ter a interação com o usuário e vai mostrar as coisas na tela**. Quando a view precisar fazer qualquer lógica ou operação, ela vai chamar o **controller,** é **lá que vão ficar os códigos que vão fazer as requisições para a APIs** por exemplo. Se seu código tiver alguma entidade, seja ela uma Pessoa Usuária, um Filme (em um site de streaming por exemplo), um Jogo, ou qualquer coisa que precise de um modelo para ser representada, ela ficará na **Model**. E quem chama a Model? O Controller! O **Controller é quem vai gerenciar todo fluxo de informações entre a View, a Model e APIs.**

Para ajudar

Links interessantes:

- O que é programação funcional e o que isso tem a ver com o Nubank?, Nubank <https://blog.nubank.com.br/programacao-funcional-o-que-e-relacao-nubank/> Acesso em ago. 2022
- Uma visão muito breve sobre o paradigma funcional, Sergio Costa, medium <https://sergiocosta.medium.com/paradigma-funcional-3194924a8d20> Acesso em ago. 2022
- Programação funcional, wikipédia <https://pt.wikipedia.org/wiki/Programa%C3%A7%C3%A3o_funcional> Acesso em ago. 2022
- Hoisting, MDN <https://developer.mozilla.org/pt-BR/docs/Glossary/Hoisting> Acesso em ago. 2022

- Hoisting no Javascript, Alura <https://www.alura.com.br/artigos/hoisting-no-javascript> Acesso em ago. 2022
- Closures, MDN <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Closures> Acesso em ago. 2022
- JavaScript Array map() <https://www.w3schools.com/jsref/jsref_map.asp> Acesso em ago. 2022