Dia 5 - Node.js

Tipos estruturados

Centro de Alta Performance - SECITECI

<Fic_Dev>
Programador de Sistemas

Roteiro

1. Desafio
2. Objetivo da aula
3. Tipos estruturados
   - Array
   - Função
   - Objetos
   - JSON
4. Operações sobre array
5. Operações sobre objetos

<Fic_Dev>
Programador de Sistemas

Desafio

Problema: Registro de Alunos
Contexto: Você está desenvolvendo um sistema de registro de alunos para uma escola. O sistema deve permitir a criação de perfis individuais para cada aluno, incluindo informações como nome, idade, curso, data de nascimento, notas e média final.

Desafio

Problema: Registro de Alunos
Descrição do problema: Você precisa desenvolver uma função que permita o registro de informações de um aluno no sistema. A função deve solicitar ao usuário que insira os seguintes dados:

1. Nome do aluno
2. Idade do aluno
3. Curso
4. Data de Nascimento
5. Notas do aluno em 4 disciplinas diferentes (por exemplo, Matemática, Ciências, História e Geografia)

<Fic_Dev>
Programador de Sistemas

Desafio

Problema: Registro de Alunos
Com base nas notas fornecidas, o programa deve calcular a média final do aluno e exibir todas as informações registradas na tela.
Requisitos do problema:

1. O programa deve solicitar ao usuário que insira o nome do aluno.
2. O programa deve solicitar ao usuário que insira a idade do aluno.
3. O programa deve solicitar ao usuário que insira as três notas do aluno.
4. O programa deve calcular a média final do aluno com base nas notas fornecidas.
5. O programa deve exibir todas as informações registradas, incluindo nome, idade, notas individuais e média final.

<Fic_Dev>
Programador de Sistemas

Objetivo da disciplina

Conhecer tipos de dados estruturados.

Entender o uso de tipos de dados estruturados.

Saber manipular tipos estruturados de dados.

Tipos estruturados

- Os tipos de dados estruturados são aqueles que são definidos pelo usuário ou são objetos.
- Eles são mutáveis, o que significa que seus valores podem ser alterados depois de definidos.
- Os tipos de dados não primitivos em JavaScript são:
  - Array
  - Objetos
  - JSON
  - Funções

Array

- Arrays são objetos usados para armazenar vários valores em uma única variável.
- Em JavaScript, os arrays são uma coleção ordenada de elementos.
- Os arrays podem conter valores de diferentes tipos, como números, strings e até mesmo outros arrays.
- Os índices dos arrays são baseados em zero, ou seja, o primeiro elemento tem índice 0, o segundo tem índice 1, e assim por diante.

Exemplos - Array

```
let numeros = [1, 2, 3];
let nomes = ["João", "Maria", "Pedro"];
let misto = [[1, 2, 3], ["João", "Maria"], 2, "Marcos"];
```

Funções

- São usadas para agrupar um conjunto de ações em um bloco de código reutilizável. Podem ser declaradas com ou sem parâmetros.

Exemplos - Funções

```
function saudacao(nome) {
    console.log("Olá, " + nome + "!");
}
saudacao("João"); // imprime "Olá, João!"
```

Objetos

- São coleções de pares de chave-valor.
- Cada chave é uma string e o valor pode ser qualquer tipo de dado, incluindo outros objetos.
- Para criar um objeto em JavaScript, você pode usar a sintaxe de objeto literal, que é colocar as chaves e valores dentro de chaves :

Exemplos - Objetos

```
const pessoa = {
    nome: "João",
    idade: 30,
    endereco: {
        rua: "Av. Paulista",
        numero: 100,
        cidade: "São Paulo"
    }
};
```

- O JSON em si não é um tipo estruturado de JavaScript.
- É um formato de dados que pode ser usado por qualquer linguagem de programação que possa analisar texto.

Exemplos - JSON

OBJETO

```
const pessoa = {
    nome: "João",
    idade: 30,
    endereco: {
        rua: "Rua A",
        numero: 100,
        cidade: "Cuiabá"
    }
};
```

JSON

```
{
    "nome": "João",
    "idade": 30,
    "endereco": {
        "rua": "Rua A",
        "numero": 100,
        "cidade": "Cuiabá"
    },
}
```

<Fic_Dev>
Programador de Sistemas

Operações sobre array

- Para criar um array em JavaScript, pode-se usar a notação de colchetes ([]):

```
let nomes = ["João", "Maria", "José"];
```

- Para acessar um elemento específico do array, utiliza-se o índice do elemento desejado:

```
// Retorna "João"
let primeiroNome = nomes[0];
```

Operações

- É possível modificar elementos do array atribuindo um novo valor ao índice desejado:

```
// Altera o segundo elemento para "Mariana"
nomes[1] = "Mariana";
```

- Outras operações: Adicionar elementos, remover elementos e encontrar o tamanho do array....

Laço de repetição em Arrays

- É comum percorrer os elementos de um array usando um laço de repetição, como o *for* ou o *for...of*.
- Exemplo de uso do *for* para percorrer um array:

```
let numeros = [1, 2, 3, 4, 5];
for (let i = 0; i < numeros.length; i++) {
  console.log(numeros[i]);
}
```

Laço de repetição em Arrays

- Exemplo de uso do *for...of* para percorrer um array:

```
let nomes = ["João", "Maria", "José"];

for (let nome of nomes) {
  console.log(nome);
}
```

Manipulando objetos

- Acessando propriedades

```
console.log(pessoa.nome); // João
let propriedade = "idade";
console.log(pessoa[propriedade]); // 30
```

- Alterando propriedades

```
pessoa.nome = "Maria";
```

- Adicionando propriedades

```
pessoa.telefone = "(11) 1234-5678";
```

- Removendo propriedades

```
delete pessoa.telefone;
```

<Fic_Dev>
Programador de Sistemas

Coleções de pacotes

- O uso de coleções dos pacotes JavaScript depende do pacote específico que você está utilizando.
- Instalado usando gerenciador *npm* e importado no seu arquivo JavaScript usando a palavra-chave *require* ou *import*.

```
npm install immutable
```

```
import { Map } from 'immutable';
const map1 = Map({ a: 1, b: 2, c: 3 });
console.log(map1);
```

Manipulando Map

- Criando novas instâncias

```
const map2 = map1.set('b', 50);
// Output: Map { "a": 1, "b": 50, "c": 3 }
console.log(map2);

const map3 = map2.delete('a');
// Output: Map { "b": 50, "c": 3 }
console.log(map3);
```

Vamos ao desafio !