Programação Mobile
React Native - Aula 05

Professor: João Felipe Bragança

Objetivos da Aula

- Tópicos de JS
- Destructuring
- Spread Operator
- Diferenças entre for, foreach e map
- Utilizando o filter

Destructuring

- A desestruturação é uma maneira conveniente de extrair valores de arrays ou objetos e atribuí-los a variáveis individuais de forma mais concisa. Isso é especialmente útil quando você tem dados estruturados em arrays ou objetos e deseja extrair partes específicas para uso posterior.

```
const person = { name: 'John', age: 30, city: 'New York' };
const { name, age } = person;

console.log(name); // 'John'
console.log(age);  // 30
```

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Operators/Destructuring_assignment

Spread Operator

- O operador spread (...) é usado para copiar elementos de arrays ou propriedades de objetos de forma rápida e conveniente.

```
const obj1 = { a: 1, b: 2 };
const obj2 = { ...obj1, c: 3 };

console.log(obj2); // { a: 1, b: 2, c: 3 }
```

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Operators/Spread_syntax

for

- É uma estrutura de repetição que permite iterar sobre os elementos de um array (ou qualquer iterável) especificando explicitamente as condições de inicialização, condição de continuação e incremento.

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

foreach

- O método forEach é uma função disponível em arrays em JavaScript. Ele permite executar uma função de callback para cada elemento do array.

```
const array = [1, 2, 3, 4, 5];
array.forEach(item => {
    console.log(item);
});
```

map

- O método map é semelhante ao forEach, mas retorna um novo array com base nos resultados da função de callback aplicada a cada elemento do array original.

```
const array = [1, 2, 3, 4, 5];
const newArray = array.map(item => {
    return item * 2;
});
console.log(newArray); // Output: [2, 4, 6, 8, 10]
```

Quando usar cada um?

- Use ***for*** quando precisar de controle detalhado sobre o loop.

- Use ***forEach*** quando quiser percorrer os elementos de um array sem a necessidade de criar um novo array.

- Use ***map*** quando quiser criar um novo array baseado nos elementos de um array original e aplicar alguma operação a cada elemento.

Em React e React Native, o map pode ser utilizado para renderizar dinamicamente uma lista na tela, basta chamá-lo no jsx entre {} e retornar algum elemento visual, por exemplo:

{ array.map(tem => <Text>{item}</Text>) }

Filter

- O método filter() é usado para criar um novo array com todos os elementos que passam por um teste (ou condição) especificado em uma função de callback. Em outras palavras, filter() é usado para filtrar elementos de um array com base em uma determinada condição.

```
const numbers = [1, 2, 3, 4, 5];
const evenNumbers = numbers.filter(num => num % 2 === 0);

console.log(evenNumbers); // Output: [2, 4]
```

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

Informações Importantes

- Já sabemos que os states só podem ser alterados pela função de setState, então não podemos alterar diretamente um array.

- Geralmente criamos um array novo com as modificações e aí sim, passamos esse novo array para o setState.

- map e filter são muito utilizados pois geram arrays novos.

- map é muito utilizado para criar listas dinâmicas no jsx.

- filter é muito utilizado para fazer qualquer tipo de filtro em um array.

Obrigado(a)!