Unyleya
EDUCACIONAL

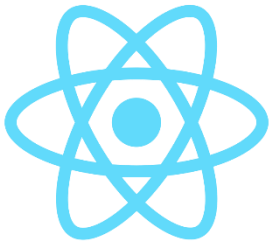
(1.1.1) Introdução ao desenvolvimento multiplataforma com React Native

O que é React Native

React Native é uma biblioteca desenvolvida pela equipe do Facebook, que possibilita ao desenvolvedor de aplicações mobile criar aplicativos para Android e iOS utilizando o JavaScript.

Vantagens

- Código único para aplicativos Android e iOS;
- Alta compatibilidade com o hardware do dispositivo;
- Reuso de código;
- Desempenho;
- Baixo custo;
- Código nativo.



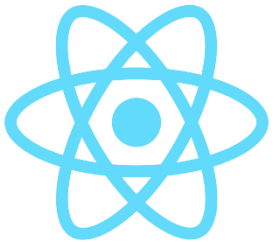
React Native é nativo! O que é isso?

Aplicações nativas são construídas em uma linguagem exclusiva para um determinado sistema operacional. O React Native converte todo o código desenvolvido para a linguagem nativa do sistema operacional do seu aparelho.

Mas todas os frameworks do mercado não fazem isso? Não!!!!!

Web App: É um site desenvolvido exclusivamente para dispositivos móveis. Possui uma programação que reconhece que o usuário está acessando por um smartphone e se adapta a ele.

Aplicativo Híbrido: O aplicativo híbrido, diferente do nativo, não foi desenvolvido completamente na linguagem específica de cada sistema operacional, o app utiliza várias linguagens e engloba dois formatos: é metade nativo e metade web app.



Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL

Unyleya
EDUCACIONAL

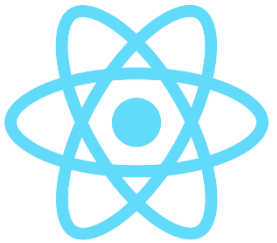
(1.2.1) Introdução ao JavaScript e Tipos de Dados

JavaScript para React Native

O React Native faz uso do JavaScript (JS) para construir seus aplicativos. O JS é uma linguagem de programação que possibilita a construção de páginas dinâmicas.

Revisão

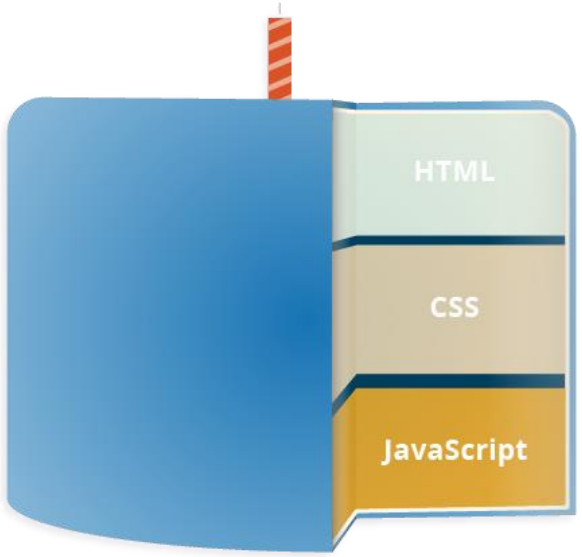
- Variáveis;
- Estrutura condicional;
- Estrutura de repetição;
- Funções.



JavaScript para React Native – Introdução

“JavaScript é uma linguagem de programação que permite a você implementar itens complexos em páginas web — toda vez que uma página da web faz mais do que simplesmente mostrar a você informação estática — mostrando conteúdo que se atualiza em um intervalo de tempo, mapas interativos ou gráficos 2D/3D animados, etc. — você pode apostar que o JavaScript provavelmente está envolvido.”

Mozilla Developers, 2020.



JavaScript para React Native – Introdução

- Variáveis são case sensitive:

```
var meunome = "Wesley";
var Meunome = "Wesley";
```

- As aspas

```
var meuNome = "Wesley";
var meuNome = 'Wesley';
```

- Comentários

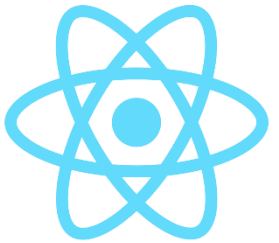
```
//var meunome = "Wesley";

/* var Meunome = "Wesley";
   var minhaIdade = 18; */
```

- O ponto e vírgula

```
var meuNome = "Wesley Silva"
var meuNome = 'Wesley Silva';
```

- O console:

```
var meunome = "Wesley Silva";

console.log('Olá, eu me chamo ' + meunome);
```



JavaScript para React Native – Variáveis

Uma variável é um objeto (uma posição, frequentemente localizada na memória) capaz de armazenar um valor ou uma expressão.

- Em JavaScript utilizamos a palavra **var** para criar uma variável**:**

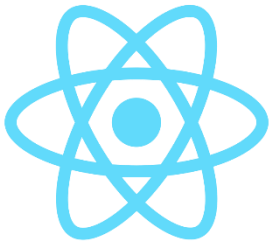
```
var minhaVariavel = "Hello World!";
```

- O ECMAScript 6, sugere utilizar as palavras **const** e **let** para criar variáveis.

```
let minhaVariavel = "Hello World!";
const minhaVariavel = "Hello World!";
```

- JS não é uma linguagem fortemente tipada.

```
let minhaVariavel = "Hello World!";
let minhaVariavel = 10;
let minhaVariavel = true;
```



JavaScript para React Native – Variáveis

Os tipos de dados de uma variável Javascript podem ser:

String: valor entre aspas simples ou aspas duplas;

```
let nome = “Wesley Silva";
```

Numérico: números;

```
let matricula = 10;
let pontos = 14.5;
```

Booleano: true (verdadeiro) ou false (falso);

```
let aprovado = true;
```

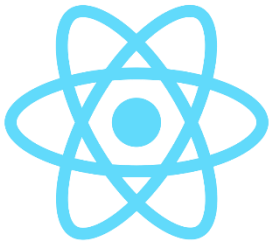
Array: veremos posteriormente;

Objeto: veremos posteriormente;

Função: veremos posteriormente.

Null e Undefined

```
let semValor = null; //null
let minhaVariavel; // undefined
```



Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL

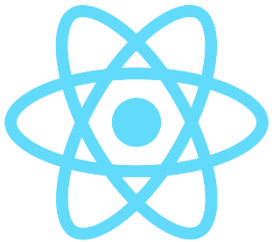
Unyleya
EDUCACIONAL

(1.2.2) Operadores Aritméticos

JavaScript para React Native – Operadores aritméticos

Operadores aritméticos tem valores numéricos (literais ou variáveis) como seus operadores e retornam um valor numérico único. Os operadores aritméticos padrões são:

- adição (+)
- subtração (-)
- multiplicação (*)
- divisão (/)



JavaScript para React Native – Operadores aritméticos

```
let numero_1 = 10;
let numero_2 = 5;

let nome = "Wesley";
let sobrenome = "Silva";

let booleano_1 = true;
let booleano_2 = false;

//numero + numero => adição
let resultado = numero_1 + numero_2; //15

//booleano + numero => adição
let resultado = booleano_1 + numero_1; //11

//booleano + booleano => adição
let resultado = booleano_1 + booleano_2; //0

//numero + string => concatenação
let resultado = numero_1 + nome; //10Wesley

//string + booleano => concatenação
let resultado = nome + booleano_1; //Wesleytrue

//string + string => concatenação
let resultado = nome + sobrenome; //WesleySilva
```

Adição (+)

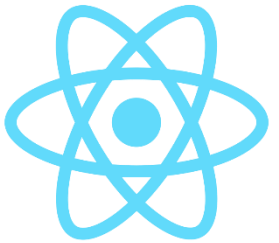
O operador de adição produz a soma dos operadores numéricos ou a concatenação de strings.

JavaScript para React Native – Operadores aritméticos

```
let numero_1 = 10;
let numero_2 = 5;

let nome = "Wesley";


//numero - numero => subtração
let resultado = numero_1 - numero_2; //5

//numero - string => NaN
let resultado = numero_1 - nome; //NaN
```

Subtração (-)

O operador de subtração subtrai os dois operandos, produzindo sua diferença.

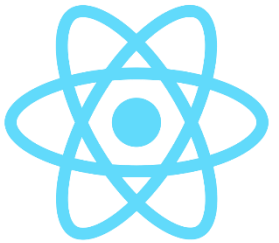


JavaScript para React Native – Operadores aritméticos

```
let numero_1 = 10;
let numero_2 = 5;

let nome = "Wesley";


//numero / numero => divisão
let resultado = numero_1 / numero_2; //2

//numero / 0 => Infinito
let resultado = numero_1 / nome; //Infinity
```

Divisão (/)

O operador de divisão produz o quociente de seus operandos onde o operando da esquerda é o dividendo e o da direita é o divisor.



JavaScript para React Native – Operadores aritméticos

```
let numero_1 = 10;
let numero_2 = 5;

let nome = "Wesley";


//numero * numero => multiplicação
let resultado = numero_1 * numero_2; //2

//Infinity * 0 => NaN
let resultado = Infinity * 0; //NaN

//Infinity * Infinity => Infinity
let resultado = Infinity * Infinity; //Infinity

//string * numero => Infinity
let resultado = nome * numero_1; //NaN
```

Multiplicação (*)

O operador de divisão produz o produto de dois ou mais valores.

Unyleya
EDUCACIONAL

Obrigado!

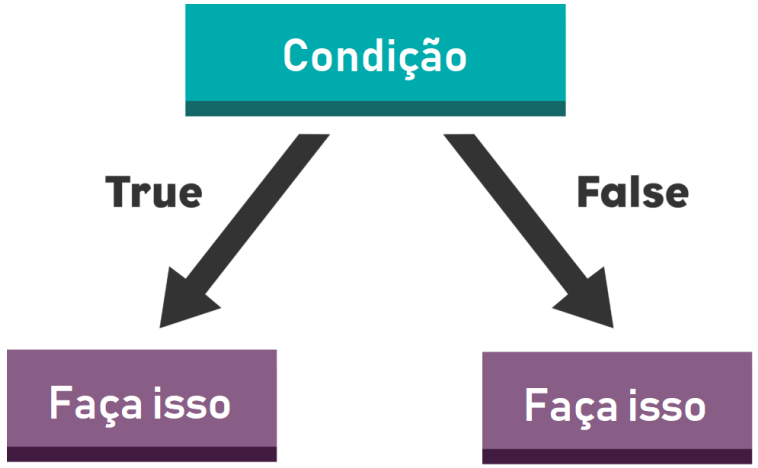
Unyleya EDUCACIONAL

Unyleya
EDUCACIONAL

(1.2.3) Estrutura condicional

JavaScript para React Native – Estrutura condicional

Uma Estrutura Condicional executa um comando ou vários comandos caso uma condição seja satisfeita.



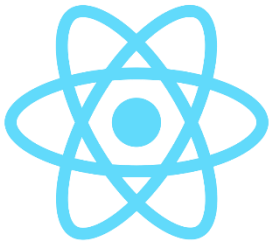
JavaScript para React Native – Estrutura condicional

Essas estruturas nos permitem tomar decisões a partir de uma escolha que deve ser feita.

Declarações if... else

```
if(condicao){
    console.log('Entrei no if');
}
else{
    console.log("Entrei no else");
}
```

```
let n1 = 5;
let n2 = 10;

if(n1 > n2){
    console.log('n1 é maior');
}
else{
    console.log('n2 é maior');
}
```



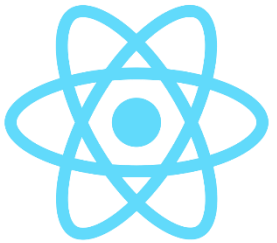
JavaScript para React Native – Estrutura condicional

Essas estruturas nos permitem tomar decisões a partir de uma escolha que deve ser feita.

Instruções Switch

```
switch(variavel){
    case valor1:
        console.log('Valor1');
        break;

    case valor2:
        console.log('Valor2');
        break;

    default:
        console.log("Algum outro valor");
}
```

```
let nome = 'Bruno';

switch(nome){

    case 'Wesley':
        console.log('Wesley');
        break;

    case 'Bruno':
        console.log('Bruno');
        break;

    default:
        console.log('Outro nome');
}
```



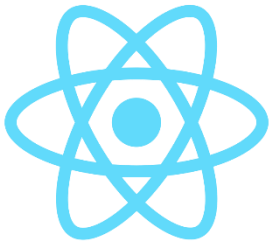
JavaScript para React Native – Estrutura condicional

Essas estruturas nos permitem tomar decisões a partir de uma escolha que deve ser feita.

Operador Ternário

```
condicao ? console.log('condição aceita') : console.log('condição não aceita');
```

```
let valor = 5;

valor > 10 ? console.log('O valor é maior que 10') : console.log('O valor não é maior que 10');
```



Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL

Unyleya
EDUCACIONAL

(1.2.4) Operadores Lógicos e comparativos

JavaScript para React Native – Operadores lógicos

Retorna um valor booleano baseado na comparação de duas ou mais condições.

Operador lógico && (AND)

```
if(condicao_1 && condicao_2){
    console.log('Hello World!');
}
```

```
let numero = 5;
let nome = "Wesley";

if(numero === 5 && nome === "Wesley"){
    console.log("Hello World");
}
```

Operador lógico || (OR)

```
if(condicao_1 || condicao_2){
    console.log('Hello World!');
}
```

```
let numero = 5;
let nome = "Wesley";

if(numero === 10 || nome === "Wesley"){
    console.log("Hello World");
}
```

Operador ! (NOT)

```
if(!false){
    console.log('Hello World!';
}
```

```
let minhaVariavel = false;

if(!minhaVariavel){
    console.log("Hello World");
}
```

JavaScript para React Native – Operadores comparativos

Retorna um valor booleano baseado na comparação de duas ou mais condições.

Igual (==)

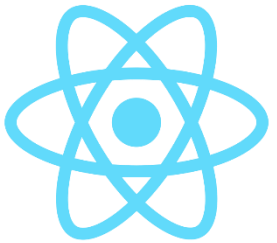
Retorna verdadeiro caso os operandos sejam iguais.

```
let var1 = 3;
let var2 = 4;

let resultado = 3 == var1; //true;
let resultado = '3' == var1; //true;
let resultado = 3 == '3'; //true;
```

Não Igual (!=)

Retorna verdadeiro caso os operandos não sejam iguais.

```
let var1 = 3;
let var2 = 4;

let resultado = var1 != 4; //true;
let resultado = var2 != '3'; //true;
```



JavaScript para React Native – Operadores comparativos

Retorna um valor booleano baseado na comparação de duas ou mais condições.

Estritamente Igual (===)

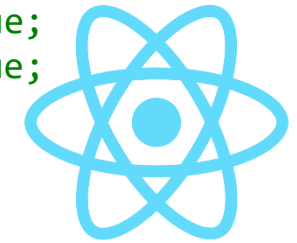
Retorna verdadeiro caso os operandos sejam iguais e do mesmo tipo.

```
let var1 = 3;
let var2 = 4;

let resultado = 3 === var1; //true;
let resultado = '3' === var1; //false;
let resultado = 3 === '3'; //false;
```

Estritamente Não Igual (!==)

Retorna verdadeiro caso os operandos não sejam iguais e/ou não sejam do mesmo tipo.

```
let var1 = 3;
let var2 = 4;

let resultado = var1 !== 4; //true;
let resultado = var2 !== '3'; //true;
let resultado = var1 !== '3'; //true;
```



JavaScript para React Native – Operadores comparativos

Retorna um valor booleano baseado na comparação de duas ou mais condições.

Maior que (>)

Retorna verdadeiro caso o operando da esquerda seja maior que o da direita.

```
let numero1 = 4;
let numero2 = 3;
let resultado = numero1 > numero2; //true
```

Maior que ou igual (>=)

Retorna verdadeiro caso o operando da esquerda seja maior que ou igual o da direita.

```
let numero1 = 4;
let numero2 = 3;
let resultado = numero1 >= numero2; //true
```

Menor que (<)

Retorna verdadeiro caso o operando da esquerda seja menor que o da direita.

```
let numero1 = 4;
let numero2 = 3;
let resultado = numero2 < numero1; //true;
```

Menor ou igual que (<=)

Retorna verdadeiro caso o operando da esquerda seja menor que ou igual o da direita.

```
let numero1 = 4;
let numero2 = 3;
let resultado = numero2 <= numero1; //true;
```

Unyleya
EDUCACIONAL

Obrigado!

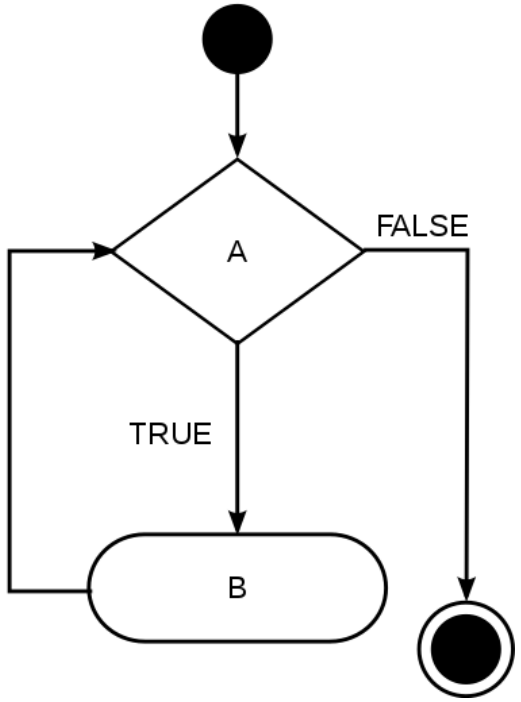
Unyleya EDUCACIONAL

Unyleya
EDUCACIONAL

(1.2.5) Estrutura de Repetição

JavaScript para React Native – Estrutura de Repetição

É uma estrutura que permite executar mais de uma vez o mesmo comando ou conjunto de comandos, de acordo com uma condição ou com um contador.



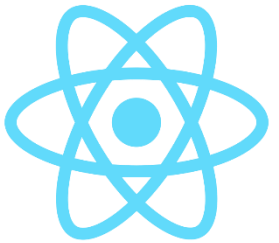
JavaScript para React Native – Estrutura de Repetição

É uma estrutura que permite executar mais de uma vez o mesmo comando ou conjunto de comandos, de acordo com uma condição ou com um contador.

for

```
for(inicialização; condição; expressão){
    console.log('Hello World!');
}
```

```
for(contador = 0; contador < 3 ; contador = contador + 1){
  console.log('Hello World!');
}
```



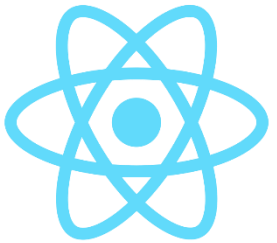
JavaScript para React Native – Estrutura de Repetição

É uma estrutura que permite executar mais de uma vez o mesmo comando ou conjunto de comandos, de acordo com uma condição ou com um contador.

while

```
while(condição){
    console.log('Hello World!');

    expressão;
}
```

```
let contador = 0;

while( contador < 3){
  console.log('Hello World!');

  contador = contador + 1; // contador++ ou contador+=1;
}
```



Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL

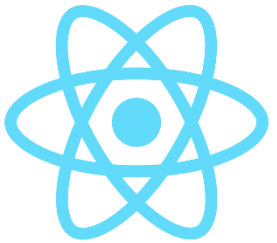
Unyleya
EDUCACIONAL

(1.3.1) Funções

JavaScript para React Native – Funções

Uma função é um subprograma que executa uma determinada tarefa.

```
function helloWorld(){
    console.log("Hello, World!");
}
```



JavaScript para React Native – Funções

Uma função é um subprograma que executa uma determinada tarefa.

function

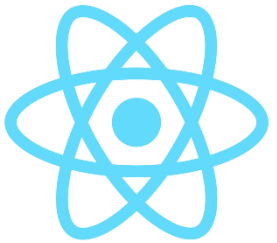
```
function minhaFuncao(parametro){
    console.log(parametro);
}

minhaFuncao(10)
```

JavaScript para React Native – Funções

Uma função é um subprograma que executa uma determinada tarefa.

Arrow function

```
const minhaFuncao = (parametro) => {
    console.log(parametro)
}

minhaFuncao(10);
```

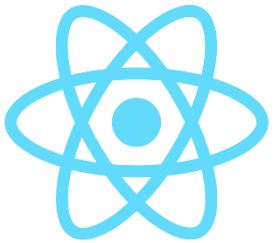


JavaScript para React Native – Funções

Callback é uma função passada como parâmetro para outra função

Callback

```
function digaOla(){
    console.log('Olá');
}


function minhaFuncao(callback){
    digaOla;
}

minhaFuncao(digaOla);
```



Unyleya
EDUCACIONAL

Obrigado!

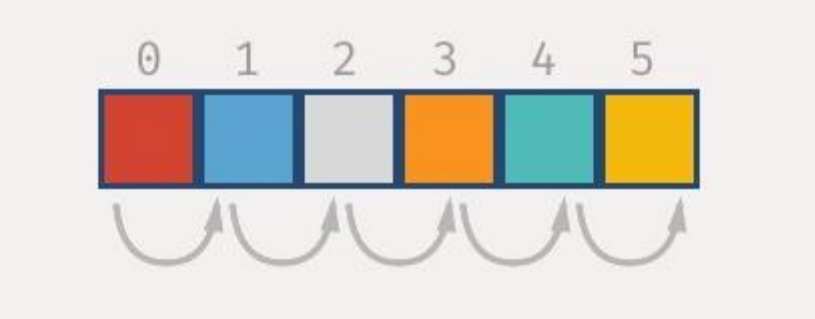
Unyleya EDUCACIONAL

Unyleya
EDUCACIONAL

(1.4.1) Arrays

Estrutura de dados para React Native – Arrays

Um array é uma sequência não ordenada de dados. Os elementos residem em um lugar separado na memória, e seu acesso é feito por meio de um índice.

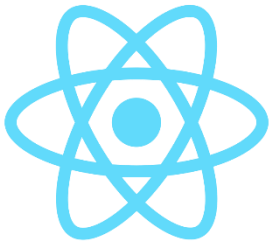


Estrutura de dados para React Native – Arrays

Um array é uma sequência não ordenada de dados. Os elementos residem em um lugar separado na memória, e seu acesso é feito por meio de um índice.

Array

```
var frutas = ['Maçã', 'Banana', 'Laranja', 'Melancia'];
var primeiraFruta = frutas[0]; // Maça
var terceiraFruta = frutas[2]; // Laranja

console.log(primeiraFruta); //Maça
```

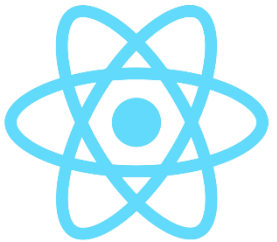


Estrutura de dados para React Native – Arrays

Um array é uma sequência não ordenada de dados. Os elementos residem em um lugar separado na memória, e seu acesso é feito por meio de um índice.

Funções básicas dos Arrays:

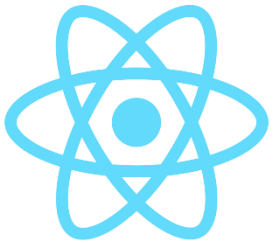
- map(): O map executa uma função em todos os elementos de um array, faz uma transformação nesses elementos e retorna um novo array com os novos valores.
- filter(): O filter executa uma função em todos os elementos de um array, e baseado em alguma condição, retorna um novo array com os valores que satisfazem essa condição.
- reduce(): O reduce executa uma função para cada elemento do array, resultando num único valor de retorno.



Estrutura de dados para React Native – Arrays

map(): O map executa uma função em todos os elementos de um array, faz uma transformação nesses elementos e retorna um novo array com os novos valores.

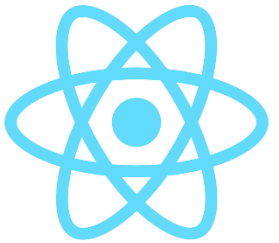
```
var numeros = [0, 1, 2, 3, 4, 5];

var arrayMultiplicado = numeros.map(function (item){
    return item * 2;
})

console.log(arrayMultiplicado); //0, 2, 4, 6, 8, 10
```



Estrutura de dados para React Native – Arrays

filter(): O filter executa uma função em todos os elementos de um array, e baseado em alguma condição, retorna um novo array com os valores que satisfazem essa condição.

```
var numeros = [0, 1, 2, 3, 4, 5];

var arrayFiltrado = numeros.filter(function (item){
    return item > 2;
})

console.log(arrayFiltrado); //3, 4, 5
```



Estrutura de dados para React Native – Arrays

reduce(): O reduce executa uma função para cada elemento do array, resultando em um único valor de retorno.

```
var numeros = [0, 1, 2, 3, 4, 5];

var arraySomado = numeros.reduce(function (item, acumulador){
    return acumulador + item;
},0)

console.log(arraySomado); //15
```

Estrutura de dados para React Native – Arrays

Mais alguns métodos básicos de arrays.

- *pop():* remove o último elemento de um array.
- *push():* adicionar um elemento no final do array.
- *unshift():* adiciona um elemento no início do array.

```
var numeros = [0, 1, 2, 3, 4, 5];
numeros.pop();
console.log(numeros); //0, 1, 2, 3, 4
```

```
var numeros = [0, 1, 2, 3, 4, 5];
numeros.push(20);
console.log(numeros); //0, 1, 2, 3, 4, 5, 20
```

```
var numeros = [0, 1, 2, 3, 4, 5];
numeros.unshift(10);
console.log(numeros); //20, 0, 1, 2, 3, 4, 5
```

Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL

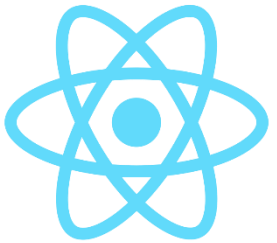
Unyleya
EDUCACIONAL

(1.5.1) Objetos

Estrutura de dados para React Native – Objetos

Objetos são caracterizados por atributos e métodos. Atributos são as propriedades de um objeto. Métodos são as ações que um objeto pode realizar.

```
var pessoa = {
    disciplina: 'React Native',
    mensagem: function(){
        alert(‘Estou aprendendo React Native!');
    }
}
```

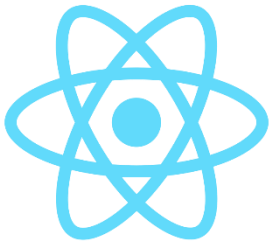


Estrutura de dados para React Native – Objetos

Objetos são caracterizados por atributos e métodos. Atributos são as propriedades de um objeto. Métodos são as ações que um objeto pode realizar.

Objetos

```
var pessoa = {
    nome: 'Ana Maria',
    idade: 32,
    sexo: 'Feminino',
    interesses: ['Música', 'Filmes'],
    saudacao: function(){
        alert('Olá, sou a Ana Maria!');
    }
}


var nome = pessoa.nome; //Ana Maria
pessoa.saudacao();
```



Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL

Unyleya
EDUCACIONAL

(1.5.2) Desconstruindo Arrays e Objetos

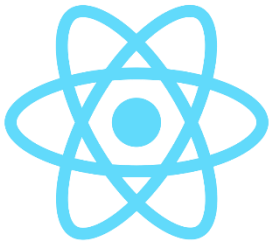
Estrutura de dados para React Native - Operators

Spread Operator: Permite expandir elementos de um array.

```
const a = [1, 2, 3];
const b = [4, 5, 6];

const c = [...a, ...b];

console.log(c); //1, 2, 3, 4, 5, 6
```

Rest Operator: Nos permite representar um número indefinido de argumentos como um array.

```
function minhaFuncao(...parametros){
    console.log(parametros);
}

console.log(minhaFuncao(1,2,3)); // 1, 2, 3
```



Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL

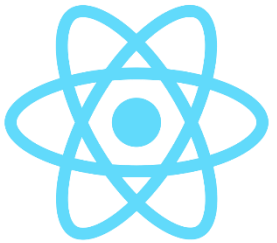
Unyleya
EDUCACIONAL

(1.6.1) JavaScript Assíncrono

JavaScript Assíncrono

JavaScript Assíncrono é um conceito aplicado para o retorno não imediato de um resultado esperado para o futuro.

```
let minhaVariavel = false;

//função para fazer requisição get
axios.get('/user/12345')

if(!minhaVariavel){
    console.log("Hello World");
}
```

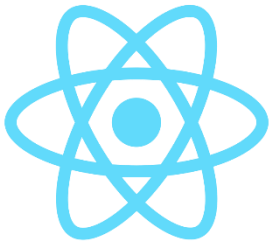


JavaScript Assíncrono

Promise é um objeto usado para processamento assíncrono. Um **Promise** (de "promessa") representa um valor que pode estar disponível agora, no futuro ou nunca.

Promises

```
function requisitarBD(){

    return new Promise((resolve, reject)=>{
        resolve('OK');
        reject('Não OK');
    });

}

requisitarBD()
    .then(resposta=>{
        console.log(resposta);
    })
    .catch(erro=>{
        console.log(erro)
    })
```



JavaScript Assíncrono

O **Async/Await** depende de uma promise para funcionar. A expressão await faz a execução de uma função async pausar, para esperar pelo retorno da Promise.

Async/Await

```
async function requisitarBD(){

    await salvarUsuario();

    return ('Usuário salvo com sucesso!');
}


requisitarBD();
```

Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL

Unyleya
EDUCACIONAL

(1.7.1) Configurando ambiente React Native

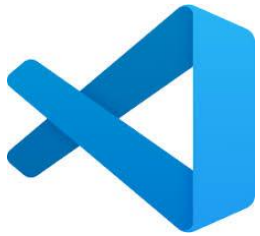
Configurando Ambiente React Native

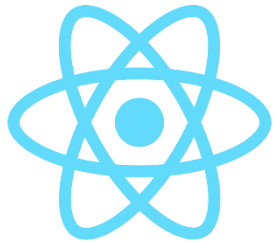
1 – Instalar Visual Studio Code (ou outro);
2 – Instalar NodeJS;
3 – Instalar JDK;
4 – Instalar Xcode (macOs);
5 – Configurar Xcode (macOs);
6 – Instalar Android Studio;
7 – Configurar Emulador;
8 – Configurar variáveis de ambiente

Principais comandos do React Native

- **Criar projeto:** npx react-native init nome_projeto
- **Executar projeto:**
  - **Android:** npx react-native run-android
  - **iOS:** npx react-native run-ios

*** Não é mais necessário instalar o react-native-cli (npm i –g react native-cli)





Referências Bibliográficas

Mozila. **JavaScript**. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript. Acesso em 19 de jun. 2020.

React Native**. A framework for building native apps using React.** Disponível em: https://reactnative.dev/. Acesso em 19 de jun. 2020.

Unyleya
EDUCACIONAL

Obrigado!

Unyleya EDUCACIONAL