



# Sintaxe e componentes do React Native

# Tema 2: Sintaxe e Componentes do React Native

## **Descrição**

Apresentação dos conceitos, das técnicas e das ferramentas ligadas ao desenvolvimento de aplicativos mobile para a plataforma Android com o uso do framework React Native.

## **Propósito**

Compreender as especificidades da programação para dispositivos móveis nos sistemas Android mediante a utilização da biblioteca React Native, baseada, por sua vez, na sintaxe de uma linguagem popular, como o JavaScript.

## **Preparação**

Para acompanhar o conteúdo e a codificação dos exemplos a serem apresentados ao longo deste estudo, você terá de utilizar uma IDE (sigla de Integrated Development Environment), sendo recomendado, para tal, o software gratuito Visual Studio Code. Além disso, você precisa configurar o ambiente de desenvolvimento e de testes no qual diferentes configurações e ferramentas podem ser usadas. Para mais detalhes a respeito dessa etapa, visite o site oficial do React Native.

## Tema 2: Sintaxe e Componentes do React Native

No início do estudo de uma linguagem de programação ou do processo de desenvolvimento de um software/aplicativo, devemos introduzir alguns conceitos relacionados ao framework React Native.

O React Native é uma biblioteca Javascript integrante do ecossistema de outro framework, o React.js. Apesar de ambos terem sido criados pelo Facebook e compartilharem algumas semelhanças, eles possuem utilizações distintas.



# Tema 2: Sintaxe e Componentes do React Native

## Criando um aplicativo móvel

O Expo é um framework e uma plataforma composta por um conjunto de ferramentas e serviços que facilita as tarefas de desenvolvimento, construção e implantação de aplicativos Android, iOS e web. Ele possui como base um mesmo código JavaScript/TypeScript.

O primeiro passo para fazer uso do Expo — e que também é o ponto de partida de tudo relacionado ao React Native, como veremos em breve — consiste na instalação de um gerenciador de pacotes. Entre suas principais opções, destacam-se o NPM e o YARN. No entanto, falaremos ainda de outro gerenciador: o NODE.JS.

Node.js é uma plataforma que permite executar código JavaScript fora do navegador, no lado do servidor. Ele é gratuito, de código aberto e multiplataforma.

### O que o Node.js pode fazer?

- Criar aplicações web
- Criar servidores
- Criar ferramentas de linha de comando
- Criar programas de automação de tarefas
- Criar APIs
- Criar sistemas em tempo real

### Como o Node.js funciona?

- Interpreta o código JavaScript, semelhante ao navegador
- Converte o código JavaScript para a linguagem de máquina que o computador executa
- Usa uma única thread para atender várias solicitações simultâneas
- Usa uma arquitetura assíncrona, não bloqueante e orientada a eventos

Saiba mais sobre o NODE.JS em <https://nodejs.org>

# Tema 2: Sintaxe e Componentes do React Native

Todo projeto que utiliza o NODE.JS possui um arquivo fundamental: "package.json". Esse é um arquivo de configuração que traz todas as informações do projeto, versão do node em uso, dependências (bibliotecas externas) e scripts (comandos). Ele é uma espécie de manifesto que descreve o aplicativo, os módulos e os pacotes utilizados. A partir dele é possível instalar TODOS os recursos necessários do projeto.

Um cenário muito comum é disponibilizar - diretamente ou por meio de um repositório/versionador (GitHub) - apenas os códigos-fonte do aplicativo ao lado do arquivo de configuração. Aliás, essa é a opção indicada, uma vez que os pacotes usados no projeto podem consumir bastante espaço em disco. Com isso, bastará a quem for utilizar nosso código baixar o projeto e executar o comando “**npm install**” para que todas as dependências sejam instaladas e o aplicativo esteja funcional.

## Gerenciadores de pacotes

### YARN

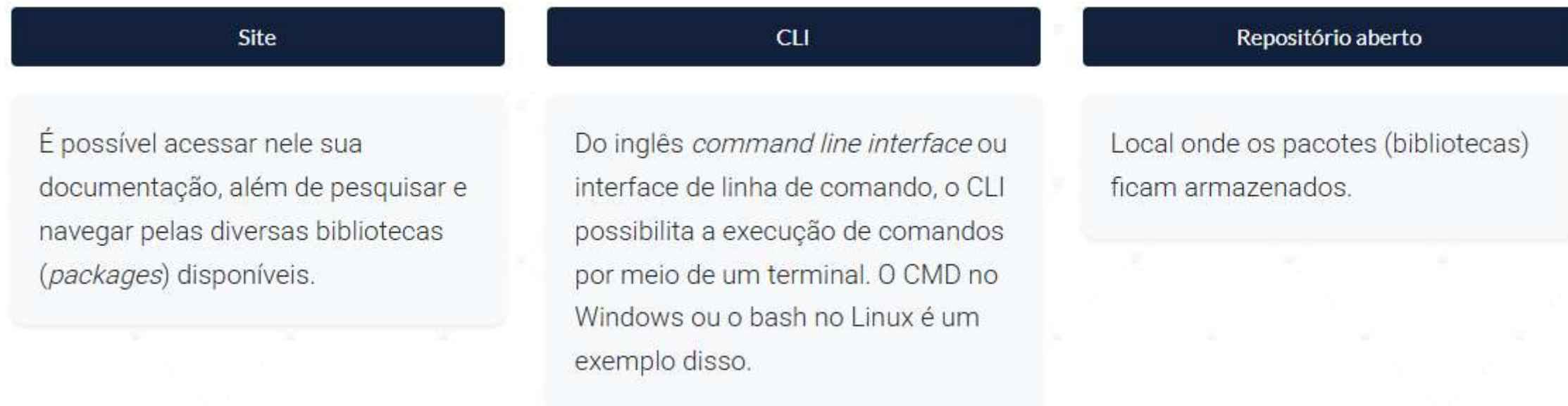
O YARN (sigla de yet another resource negotiator) foi lançado em 2016 pelo Facebook com outras empresas, dentre elas, a Google. Sua criação teve como premissa resolver alguns problemas de segurança existentes no NPM à época, além de tornar mais rápido o processo de instalação de dependências.

# Tema 2: Sintaxe e Componentes do React Native

## NPM

O NPM (acrônimo de node package manager) é um gerenciador de pacotes lançado no biênio 2009-2010. Tal pacote faz parte da instalação padrão do ambiente de execução da linguagem JavaScript Node.js, sendo ambos instalados de forma conjunta.

O NPM possui três componentes:



# Tema 2: Verificando o aprendizado

## Questão 1

Os gerenciadores de pacotes/dependências possuem um importante papel no desenvolvimento de aplicativos. Com base nisso, escolha a alternativa correta.

[A] Embora os gerenciadores de pacote sejam importantes, existe a opção de realizar as tarefas desempenhas por eles de forma manual. Isso nos permite um maior controle sobre o que é instalado em nossos ambientes de desenvolvimento.

[B] Os gerenciadores de pacote para desenvolvimento na plataforma Android só estão disponíveis nativamente no sistema operacional Linux. No Windows, é preciso instalar o Java para ter acesso aos gerenciadores de pacote.

[C] Os gerenciadores facilitam a gestão (instalação, controle de versão etc.) de pacotes. Entretanto, em relação ao React Native, realizar a instalação deles se torna uma tarefa complicada independentemente do sistema operacional utilizado, já que há apenas um gerenciador disponível, assim como uma única forma de instalação.

[D] Os gerenciadores NPM e YARN desempenham o mesmo papel. Logo, não havendo nenhum tipo de restrição ou recomendação específica inerente a determinado projeto específico, o desenvolvedor poderá optar por um dos dois. O único cuidado é evitar o uso de ambos num mesmo projeto.

[E] O YARN, lançado posteriormente ao NPM, além de mais seguro, conta com uma variedade maior de pacotes disponíveis.

# Tema 2: Verificando o aprendizado

## Questão 1

Os gerenciadores de pacotes/dependências possuem um importante papel no desenvolvimento de aplicativos. Com base nisso, escolha a alternativa correta.

[A] Embora os gerenciadores de pacote sejam importantes, existe a opção de realizar as tarefas desempenhas por eles de forma manual. Isso nos permite um maior controle sobre o que é instalado em nossos ambientes de desenvolvimento.

[B] Os gerenciadores de pacote para desenvolvimento na plataforma Android só estão disponíveis nativamente no sistema operacional Linux. No Windows, é preciso instalar o Java para ter acesso aos gerenciadores de pacote.

[C] Os gerenciadores facilitam a gestão (instalação, controle de versão etc.) de pacotes. Entretanto, em relação ao React Native, realizar a instalação deles se torna uma tarefa complicada independentemente do sistema operacional utilizado, já que há apenas um gerenciador disponível, assim como uma única forma de instalação.

**[D] Os gerenciadores NPM e YARN desempenham o mesmo papel. Logo, não havendo nenhum tipo de restrição ou recomendação específica inerente a determinado projeto específico, o desenvolvedor poderá optar por um dos dois. O único cuidado é evitar o uso de ambos num mesmo projeto.**

[E] O YARN, lançado posteriormente ao NPM, além de mais seguro, conta com uma variedade maior de pacotes disponíveis.



# Tema 2: Verificando o aprendizado

## Questão2

Um dos problemas no desenvolvimento mobile, no que tange ao ambiente de trabalho, é o alto consumo de recursos de hardware. Uma alternativa para essa situação é utilizar o framework Expo. A respeito dele, marque a alternativa correta.

[A] O Expo é um framework que, para ser utilizado, depende da biblioteca React Native. Logo, para que possa ser usado, é necessário instalar primeiramente o React Native por intermédio de um gerenciador de pacotes.

[B] O Expo é uma ferramenta que facilita o desenvolvimento e o teste de aplicações escritas em React Native. Sua principal limitação é não permitir a utilização dos componentes nativos do React Native.

[C] O Expo é um framework desenvolvido pelo Facebook. Ele facilita o desenvolvimento de aplicativos em React Native, mas sua principal função é resolver problemas contidos no NPM.

[D] O Expo restringe as funcionalidades disponíveis para a plataforma Android. Ou seja, com ele, não é possível desenvolver ou testar o código React Native para a plataforma iOS.

[E] Com o Expo CLI, é possível desenvolver aplicações da mesma forma que é feita com o React Native CLI. Ou seja, pode-se acessar os mesmos recursos e funcionalidades, exceto a câmera e o microfone do dispositivo que executa a aplicação criada.

# Tema 2: Verificando o aprendizado

## Questão2

Um dos problemas no desenvolvimento mobile, no que tange ao ambiente de trabalho, é o alto consumo de recursos de hardware. Uma alternativa para essa situação é utilizar o framework Expo. A respeito dele, marque a alternativa correta.

[A] O Expo é um framework que, para ser utilizado, depende da biblioteca React Native. Logo, para que possa ser usado, é necessário instalar primeiramente o React Native por intermédio de um gerenciador de pacotes.

**[B] O Expo é uma ferramenta que facilita o desenvolvimento e o teste de aplicações escritas em React Native. Sua principal limitação é não permitir a utilização dos componentes nativos do React Native.**

[C] O Expo é um framework desenvolvido pelo Facebook. Ele facilita o desenvolvimento de aplicativos em React Native, mas sua principal função é resolver problemas contidos no NPM.

[D] O Expo restringe as funcionalidades disponíveis para a plataforma Android. Ou seja, com ele, não é possível desenvolver ou testar o código React Native para a plataforma iOS.

[E] Com o Expo CLI, é possível desenvolver aplicações da mesma forma que é feita com o React Native CLI. Ou seja, pode-se acessar os mesmos recursos e funcionalidades, exceto a câmera e o microfone do dispositivo que executa a aplicação criada.

# Tema 2: Verificando o aprendizado

## Questão2: Comentário

A alternativa "B" está correta.

O **EXPO** fornece uma série de vantagens — principalmente no início do aprendizado de desenvolvimento mobile. Entre elas, destaca-se a facilidade de instalação, de uso e de acesso a recursos, como API e hardware do dispositivo no qual a aplicação está rodando, microfone, câmera e player de música, entre outros. Por outro lado, a principal desvantagem de sua utilização é não poder acessar os componentes nativos de cada plataforma — no caso, Android e iOS.

Em outras palavras, o EXPO não permite mesclar código nativo para o dispositivo móvel com o código escrito utilizando suas bibliotecas.

# Tema 2: Sintaxe e Componentes do React Native

## MODULO 2: Componentes

Na engenharia de software, alguns conceitos são muito utilizados para se definir o que são os componentes. Tais conceitos se referem tanto aos aspectos mais técnicos quanto aos mais práticos.

Tomando como base essa segunda abordagem, ou seja, de ordem prática, podemos enxergar os componentes como insumos, artefatos ou simplesmente “coisas” que facilitam o processo de desenvolvimento, uma vez que eles tornam dispensável que uma única pessoa, equipe ou até mesmo empresa (de software) tenha de desenvolver todas as “peças” do software que está escrevendo ou que tenha que começar todo projeto do zero.

Isso ocorre por conta de uma das principais características de um componente: ser integrado por pequenos pedaços de software que desempenham uma função (ou poucas funções) específica.

Ao pensarmos na codificação de um software com base em componentes, devemos ter em mente que, em vez de sempre fazermos a construção, podemos realizar também a composição. Isto é, podemos construir pequenos pedaços de código (os componentes), os quais, quando reunidos, formarão o software como um todo. Tais princípios se aplicam a situações nas quais são desenvolvidas tanto as aplicações de back-end quanto as aplicações e/ou os aplicativos de front-end.

# Tema 2: Sintaxe e Componentes do React Native

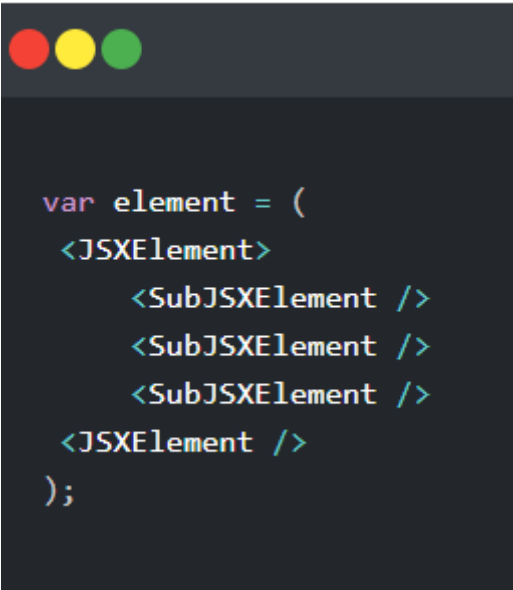
## JSX

De maneira simples e, ao mesmo tempo, completa, podemos inicialmente definir o JSX como uma sintaxe de extensão da linguagem JavaScript bastante familiar da linguagem de marcação XML.

Os componentes disponíveis em React Native são escritos utilizando JSX.

Aprofundando um pouco os conceitos, o JSX também é conhecido como JavaScript XML. Extensão semelhante ao XML para a especificação ECMAScript, ele combina a lógica de componentes (JavaScript) e o mark-up (DOM/modelo de objeto de documento ou Native UI/interface de usuário Nativa) em um único arquivo/código.

Este fragmento de código mostra a forma de um elemento JSX:



```
var element = (  
  <JSXElement>  
    <SubJSXElement />  
    <SubJSXElement />  
    <SubJSXElement />  
  </JSXElement />  
);
```

# Tema 2: Sintaxe e Componentes do React Native

## Componentes nativos

Um dos principais pilares — provavelmente o principal — do React Native é a utilização de componentes, ou seja, coleções de dados e elementos de interface gráfica que compõem as views e, de forma geral, os aplicativos em si. Embora exista a flexibilidade de desenvolver os próprios componentes customizados, o framework React Native já disponibiliza, no momento da instalação, uma série de componentes chamados de componentes nativos . Outro conceito associado a ele é o de core components.

No desenvolvimento específico para Android e iOS, as views são construídas utilizando respectivamente o Kotlin (ou Java) e o Swift (ou Objective-C). Graças ao framework React Native, é possível invocar essas views por meio dos componentes React escritos com JavaScript.

Em React, os componentes são escritos utilizando o JSX e estão agrupados em diferentes categorias. Os elementos principais estão destacados no quadro a seguir. Os componentes correspondentes em cada tecnologia que constam nela, por sua vez, serão descritos em detalhes na sequência.

# Tema 2: Sintaxe e Componentes do React Native

## Componentes nativos

Um dos principais pilares — provavelmente o principal — do React Native é a utilização de componentes, ou seja, coleções de dados e elementos de interface gráfica que compõem as views e, de forma geral, os aplicativos em si. Embora exista a flexibilidade de desenvolver os próprios componentes customizados, o framework React Native já disponibiliza, no momento da instalação, uma série de componentes chamados de componentes nativos . Outro conceito associado a ele é o de core components.

No desenvolvimento específico para Android e iOS, as views são construídas utilizando respectivamente o Kotlin (ou Java) e o Swift (ou Objective-C). Graças ao framework React Native, é possível invocar essas views por meio dos componentes React escritos com JavaScript.

Em React, os componentes são escritos utilizando o JSX e estão agrupados em diferentes categorias. Os elementos principais estão destacados no quadro a seguir. Os componentes correspondentes em cada tecnologia que constam nela, por sua vez, serão descritos em detalhes na sequência.

## Tema 2: Sintaxe e Componentes do React Native

Componente UI React Native	Componente Android	Componente iOS	Elemento HTML
<View>	<ViewGroup>	<UIView>	<div>
<Text>	<TextView>	<UITextView>	<p>
<Image>	<ImageView>	<UIImageView>	<img>
<TextInput>	<EditText>	<UITextField>	<input type="text">
<ScrollView>	<ScrollView>	<UIScrollView>	<div>



# Tema 2: Sintaxe e Componentes do React Native

## Exemplo

```
import React from "react";
import { View, Text } from "react-native";

const ViewExemplo = () => {
  return (
    <View
      style={{
        flexDirection: "row",
        height: 100,
        padding: 20
      }}
    >
      <View style={{ backgroundColor: "red", flex: 0.5 }} />
      <Text>Olá, mundo!</Text>
    </View>
  );
};

export default ViewExemplo;
```

# Tema 2: React Native

## Exemplo

```
import React, { useState } from "react";
import { Text, StyleSheet } from "react-native";

const TextoAninhado = () => {
  const [titulo, setTitulo] = useState("Texto do elemento filho");

  const modificaTexto = () => {
    setTitulo("Esse texto está sendo exibido pois o primeiro elemento de texto foi pressionado/tocado");
  };

  return (
    <Text style={styles.baseText}>
      <Text style={styles.titulo} onPress={modificaTexto}>
        {titulo}
        {"\n"}
        {"\n"}
      </Text>
    </Text>
  );
};

const styles = StyleSheet.create({
  baseText: {
    fontFamily: "Verdana",
    marginTop: 50,
    marginLeft: 10
  },
  titulo: {
    marginTop: 10,
    fontSize: 18,
    fontWeight: "bold"
  }
});

export default TextoAninhado;
```

# Tema 2: Verificando o aprendizado

## Questão 1

Um dos principais fatores que dificulta o desenvolvimento de aplicativos mobile é a diferença existente entre cada plataforma e seus respectivos sistemas operacionais. Logo, tais diferenças naturalmente fazem com que seja necessário escrever/repetir basicamente o mesmo código para atender a cada S.O. Uma solução para resolver tal problema seria

[A] fazer um movimento que envolva empresas e desenvolvedores para sensibilizar os fabricantes de dispositivos móveis a fim de que eles adotem uma plataforma em comum.

[B] escolher um sistema operacional e desenvolver aplicativos apenas para os dispositivos que o utilizem.

[C] no lugar de desenvolver aplicativos que utilizem componentes nativos, deve-se dar preferência ao desenvolvimento de websites ou de SPA (em inglês, single-page application ou, em português, aplicação web que roda em uma única página).

[D] utilizar bibliotecas que permitam a escrita de um único código-fonte, ficando ela responsável por transpilar o código criado a fim de que ele rode nos diferentes sistemas operacionais.

[E] desenvolver o mesmo aplicativo, replicando/duplicando o código-fonte para que ele seja compatível com cada sistema operacional.

# Tema 2: Verificando o aprendizado

## Questão 1

Um dos principais fatores que dificulta o desenvolvimento de aplicativos mobile é a diferença existente entre cada plataforma e seus respectivos sistemas operacionais. Logo, tais diferenças naturalmente fazem com que seja necessário escrever/repetir basicamente o mesmo código para atender a cada S.O. Uma solução para resolver tal problema seria

[A] fazer um movimento que envolva empresas e desenvolvedores para sensibilizar os fabricantes de dispositivos móveis a fim de que eles adotem uma plataforma em comum.

[B] escolher um sistema operacional e desenvolver aplicativos apenas para os dispositivos que o utilizem.

[C] no lugar de desenvolver aplicativos que utilizem componentes nativos, deve-se dar preferência ao desenvolvimento de websites ou de SPA (em inglês, single-page application ou, em português, aplicação web que roda em uma única página).

**[D] utilizar bibliotecas que permitam a escrita de um único código-fonte, ficando ela responsável por transpilar o código criado a fim de que ele rode nos diferentes sistemas operacionais.**

[E] desenvolver o mesmo aplicativo, replicando/duplicando o código-fonte para que ele seja compatível com cada sistema operacional.

# Tema 2: Verificando o aprendizado

## Comentário:

**A alternativa "D" está correta.**

O processo de desenvolvimento de aplicativos pode se tornar muito custoso caso optemos por desenvolver um mesmo código diversas vezes, adaptando-o para que ele se adeque às particularidades de cada sistema operacional, rodando, assim, nos mais diversos dispositivos móveis. Além de custoso, esse processo também dificulta a manutenção do código e sua evolução. Desse modo, utilizar uma biblioteca que possibilite escrever um único código e rodá-lo em diferentes sistemas operacionais traz inúmeros benefícios.

# Tema 2: Verificando o aprendizado

## Questão 2

O React Native possui uma série de componentes nativos, como, por exemplo, <View> e <Text>. Considerando o que vimos sobre esses componentes e analisando o fragmento de código abaixo, marque a alternativa correta quanto ao que será exibido na tela do dispositivo.

```
export default function App() {  
  return (  
    <Div style={styles.container}>  
      <Text style={styles.paragraph}>  
        Altere este texto e salve. Observe o resultado em seu telefone!  
      </Text>  
      <Card>  
        <AssetExample />  
      </Card>  
    </Div>  
  );  
}
```

# Tema 2: Verificando o aprendizado

## Questão 2

O React Native possui uma série de componentes nativos, como, por exemplo, `<View>` e `<Text>`. Considerando o que vimos sobre esses componentes e analisando o fragmento de código abaixo, marque a alternativa correta quanto ao que será exibido na tela do dispositivo.

[A] Será exibido um erro informando que a variável “Div” não existe.

[B] Será exibido, independentemente da plataforma do dispositivo, o texto contido pelo componente `<Text>`, uma vez que seu contêiner poderá ser um componente `<View>` ou o seu equivalente em HTML, como a `<Div>`.

[C] Em dispositivos Android, o texto será exibido corretamente, sem nenhum erro. Entretanto, isso não acontecerá em dispositivos iOS, já que essa plataforma não possui um componente nativo equivalente à `<Div>`.

[D] O código acima apresentará comportamento distinto se for executado no Expo ou em um dispositivo virtual/físico.

[E] Como o componente `<Div>` não faz parte dos componentes nativos do React Native, será preciso utilizar uma classe de estilos (StyleSheet) para a renderização correta do conteúdo do contêiner em questão.

# Tema 2: Verificando o aprendizado

## Questão 2

O React Native possui uma série de componentes nativos, como, por exemplo, `<View>` e `<Text>`. Considerando o que vimos sobre esses componentes e analisando o fragmento de código abaixo, marque a alternativa correta quanto ao que será exibido na tela do dispositivo.

**[A] Será exibido um erro informando que a variável “Div” não existe.**

[B] Será exibido, independentemente da plataforma do dispositivo, o texto contido pelo componente `<Text>`, uma vez que seu contêiner poderá ser um componente `<View>` ou o seu equivalente em HTML, como a `<Div>`.

[C] Em dispositivos Android, o texto será exibido corretamente, sem nenhum erro. Entretanto, isso não acontecerá em dispositivos iOS, já que essa plataforma não possui um componente nativo equivalente à `<Div>`.

[D] O código acima apresentará comportamento distinto se for executado no Expo ou em um dispositivo virtual/físico.

[E] Como o componente `<Div>` não faz parte dos componentes nativos do React Native, será preciso utilizar uma classe de estilos (StyleSheet) para a renderização correta do conteúdo do contêiner em questão.



# Tema 2: Verificando o aprendizado

## Comentário:

**A alternativa "A" está correta.**

O React Native possui alguns componentes nativos que são transpilados para os componentes equivalentes em cada plataforma onde o aplicativo é executado. Além disso, pode-se criar os próprios componentes customizados. Por outro lado, caso se utilize um componente não existente entre os nativos ou que não tenha sido criado por nós mesmos (ou seja, um importado para o projeto), a aplicação retornará um erro, informando que o elemento em questão não existe.

# Tema 2: Sintaxe e Componentes do React Native

## MODULO 3: Depuração

Uma depuração ou um debug (termo em inglês comumente utilizado na área de desenvolvimento de software) é o processo de identificar erros (bugs) ou problemas no código-fonte de um software.

Ao longo de tal processo, o código-fonte é inspecionado e analisado durante sua execução, a fim de que qualquer erro existente possa ser identificado e corrigido.

O processo de debug pode ser iniciado de duas formas:

### Indiretamente

A partir de um erro gerado na execução do aplicativo.

### Iniciativa do desenvolvedor

De maneira estruturada e previamente organizada, tal processo pretende testar o software em diferentes situações de uso.

# Tema 2: Sintaxe e Componentes do React Native

## **MODULO 3: Depuração**

- **Ferramentas de depuração de código**
  - **Inspecionar a saída com um navegador**
  - **A partir da janela do Metro, pressione a tecla “d”**
  - **Utilizando react-devtools**
- 
- **Questões**

# Referências

REACT NATIVE. Docs. Consultado na internet em: 17 de fevereiro de 2025.

<https://reactnative.dev/docs/getting-started>