Documentação para Iniciantes em React Native

O que é o React Native?

O React Native é um framework criado pelo Facebook que permite desenvolver aplicações mobile nativas usando JavaScript e React. Com ele, você pode criar apps tanto para Android quanto para iOS com uma base de código única.

Por que usar React Native?

- **Reutilização de Código**: Um único código para Android e iOS.
- **Comunidade ativa**: Muitos exemplos e bibliotecas prontas.
- **Performance nativa**: Usa componentes nativos reais.
- **Desenvolvimento rápido**: Hot Reload para ver mudanças em tempo real.

Instalando o React Native

1. **Requisitos**:
    a. **Node.js**: Runtime para executar código JavaScript fora do navegador. Essencial para usar o React Native.
    b. **npm (Node Package Manager) ou yarn**: Gerenciadores de pacotes para instalar bibliotecas e dependências.
    c. **Expo CLI**: Ferramenta que simplifica o desenvolvimento e facilita testes sem precisar configurar um ambiente nativo.
    d. **Android Studio** (opcional, mas recomendado): Excelente para testar e depurar apps Android localmente. Inclui o Android Emulator, que simula um celular no seu computador.

2. **Instalação**:

```
npm install -g expo-cli
```

3. **Criando o projeto**:

```
npx create-expo-app MeuApp
cd MeuApp
npm start
```

4. **Alternativas para desenvolvimento online**:
    a. Expo Snack: Uma plataforma online onde você pode testar código React Native diretamente no navegador, sem precisar instalar nada no seu computador.

Estrutura de um Projeto React Native

```
MeuApp/
├── App.js          # Arquivo principal
├── assets/         # Imagens, fontes etc
├── components/     # Componentes reutilizáveis
├── screens/        # Telas do app
└── package.json    # Dependências do projeto
```

Conceitos Fundamentais

1. JSX (JavaScript + XML)

JSX é uma sintaxe que parece HTML, mas que funciona dentro do JavaScript. Isso facilita a criação de interfaces visuais.

```
<View>
  <Text>Olá, mundo!</Text>
</View>
```

2. Componentes

Tudo em React Native é um **componente**.

Componentes são blocos reutilizáveis de interface. Eles podem representar desde um botão simples até uma tela inteira. Isso ajuda a organizar o código e a reaproveitar funcionalidades.

Existem dois tipos principais:

- **Componentes funcionais**: baseados em funções (mais modernos).
- **Componentes de classe**: baseados em classes (menos usados atualmente).

Exemplo de componente funcional:

```
function MeuComponente() {
  return (
    <View>
      <Text>Sou um componente!</Text>
    </View>
  );
}
```

3. Props (Propriedades)

Usadas para passar dados para componentes. É como personalizar o comportamento ou aparência de um componente.

```
function Saudacao(props) {
  return <Text>Olá, {props.nome}!</Text>;
}

// Uso:
<Saudacao nome="Maria" />
```

4. State (Estado)

Guarda informações internas do componente, que podem mudar com o tempo (ex: valor de um contador, texto digitado, etc).

```
import { useState } from 'react';

function Contador() {
  const [contador, setContador] = useState(0);

  return (
    <View>
      <Text>{contador}</Text>
      <Button title="Somar" onPress={() => setContador(contador +
```

```
1)} />
    </View>
  );
}
```

Estilizando Componentes

React Native não usa CSS tradicional. Usa objetos JavaScript para definir estilos. Os estilos são aplicados usando a propriedade `style`.

```
const styles = StyleSheet.create({
  texto: {
    fontSize: 20,
    color: 'blue',
  },
});

<Text style={styles.texto}>Texto estilizado</Text>
```

Navegação entre Telas

1. Instale:

```
npm install @react-navigation/native
expo install react-native-screens react-native-safe-area-context
react-native-gesture-handler react-native-reanimated
npm install @react-navigation/native-stack
```

2. Exemplo:

```
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-
navigation/native-stack';

const Stack = createNativeStackNavigator();
```

```
function App() {
  return (
    <NavigationContainer>
      <Stack.Navigator>
        <Stack.Screen name="Home" component={HomeScreen} />
        <Stack.Screen name="Perfil" component={PerfilScreen} />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
```

Dicas de Boas Práticas

1. **Componentização**: Separe trechos repetidos em componentes reutilizáveis.
2. **Nomes significativos**: Use nomes claros para variáveis e funções.
3. **Organização**: Separe pastas por funcionalidades.
4. **Evite lógica pesada nos componentes**: Prefira hooks ou funções externas.
5. **Use o ESLint e Prettier**: Para manter o código limpo e padronizado.
6. **Teste em ambos os sistemas**: Android e iOS têm diferenças sutis.
7. **Evite hardcode**: Sempre que possível, evite valores fixos no meio do código (como cores, textos, etc). Use constantes.

Recursos Extras

- Documentação oficial do React Native
- Expo
- React Navigation
- Snack (Ambiente Online)

Exemplo de App Completo

```
import React, { useState } from 'react';
import { View, Text, Button, StyleSheet } from 'react-native';
```

```
export default function App() {
  const [contador, setContador] = useState(0);

  return (
    <View style={styles.container}>
      <Text style={styles.titulo}>Contador</Text>
      <Text style={styles.numero}>{contador}</Text>
      <Button title="Somar" onPress={() => setContador(contador +
1)} />
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
  },
  titulo: {
    fontSize: 24,
    fontWeight: 'bold',
  },
  numero: {
    fontSize: 40,
    marginVertical: 20,
  },
});
```