Flutter - Introdução ao Desenvolvimento Móvel

Professora: Marcio

EAD 03 - Atividade de Pesquisa Comparativa: Flutter vs React Native

Data de entrega:

Aberto: 04/07/2025,13:30

Vencimento: sexta-feira, 04/07/2025, 17:30.

Integrantes do Grupo:

1. André Felipe I. Leite
2. David Luis Kim

Descrição da Atividade

“Investigar e comparar como os dois frameworks Flutter e React Native implementam quatro elementos fundamentais do desenvolvimento mobile: entrada de dados, botões, navegação entre telas e estilos visuais.”

Tarefa

Crie um quadro comparativo explicando como os seguintes quatro elementos são utilizados em Flutter e em React Native:

Elemento	Flutter	React Native
Entrada de dados	TextField	TextInput
Botões	ElevatedButton, TextButton	Button, TouchebleOpacity
Navegação entre telas	Navigator.push/Material PageRoute	react-navigation (navigate)
Estilização e layout	ThemeData, TextStyle, Container	StyleSheet, View, TextStyle

Flutter:

- Entrada de dados

  Text Field: O Flutter utiliza o widget **TextField** para receber entrada de texto do usuário. Esse widget permite a customização visual e funcional, incluindo controle de validações, máscara de texto e manipulação de foco.

```
TextField(
  decoration: InputDecoration(labelText: 'Digite seu nome'),
)

```

Print 01: Comando TextField em ação.

- Botões

  Text Button, Elevated Button: Flutter oferece múltiplas opções de botões, como **TextButton**, **ElevatedButton** e **OutlinedButton**. Cada um é usado conforme o estilo desejado: botões planos, elevados ou com borda.

```
ElevatedButton(
  onPressed: () {},
  child: Text('Enviar'),
)

```

Print 02: Exemplo de Elevated Button.

- Navegação entre telas

  Navigator.push, Material Page Route: A navegação entre telas em Flutter é feita através do **Navigator**, com push e pop, geralmente utilizando **MaterialPageRoute** para criar a transição.

```
Navigator.push(
  context,
  MaterialPageRoute(builder: (context) => NovaTela()),
);

```

Print 03: Navigator em ação utilizando o Material Page Route.

- Estilização e layout

  ThemeData, TextStyle, Container: O Flutter oferece um sistema de estilização integrado, baseado em widgets como Container, Padding, Column e Row. Estilos como TextStyle e temas globais (ThemeData) são amplamente usados.

```dart
Container(
  padding: EdgeInsets.all(16),
  child: Text(
    'Olá!',
    style: TextStyle(fontSize: 20),
  ),
)

```

Print 04: Container com padding, texto e estilo de texto.

React Native

- Entrada de dados

  TextInput: Em React Native, o componente equivalente é o TextInput. Ele também suporta personalização e eventos, como onChangeText e onSubmitEditing.

```jsx
<TextInput
  placeholder="Digite seu nome"
  onChangeText={setNome}
/>

```

Print 05: TextInput com um campo digite seu nome.

- Botões

Button, TouchableOpacity: Em React Native, o botão básico é o Button. Para mais controle visual e interação, utiliza-se TouchableOpacity, que permite criar botões personalizados com animações de toque.

```
<TouchableOpacity onPress={handlePress}>
  <Text>Enviar</Text>
</TouchableOpacity>

```

Print 06: botão para pressionar e enviar texto.

- Navegação entre telas

  react-navigation (navigate): React Native usa a biblioteca react-navigation, amplamente adotada na comunidade. A navegação é feita com **navigate**, fornecido por um Stack Navigator.

```
navigation.navigate('NovaTela');

```

Print 07: código de navegação para “NovaTela”.

- Estilização e layout

  StyleSheet, View, TextStyle: Em React Native, os estilos são definidos com o objeto **StyleSheet,** semelhante ao CSS. A construção do layout é feita com **View**, **Text**, e **Flexbox**.

```
<View style={styles.container}>
  <Text style={styles.texto}>Olá!</Text>
</View>

```

Print 08: utilizando do view.

```
const styles = StyleSheet.create({
  container: { padding: 16 },
  texto: { fontSize: 20 }
});

```

Print 09: utilizando flexbox.

Comparação entre Flutter e React Native em uma tela de login (na prática):

1. **Estrutura:**

   Flutter usa widgets aninhados (**árvore de widgets**)

   React Native usa componentes com **JSX**

2. **Gerenciamento de estado:**

   Flutter usa **setState** ou gerenciadores de estado como **Provider/Bloc**

   React Native usa hooks (**useState**)

3. **Estilização:**

   Flutter estiliza diretamente nos widgets

   React Native geralmente implementa validação manualmente

4. **Validação:**

   Flutter tem **Form** e **TextFormFieId** com validação integrada

   React Native geralmente implementa React Native geralmente

5. **Sintaxe:**

   Flutter (**Dart**) é mais verboso com forte tipagem

   React (**JavaScript/TypeScript**) é mais conciso

6. **Arquitetura:**

   Flutter separa claramente **widget** de estado (**StatefulWidget**)

   React Native usa componentes funcionais com **hooks**

Ambos criam interfaces nativas, mas o Flutter compila para código nativo enquanto o React Native usa uma ponte JavaScript para componentes nativos.