Módulo 1: Interface Interativa e Componentes Básicos

Este módulo aborda a preparação do ambiente, o planejamento da interface e os componentes fundamentais de *layout* e interação.

|  |  |  |
| --- | --- | --- |
| Ponto Importante | Detalhe | Citação |
| **Preparação do Ambiente** | Recomenda-se a IDE **Visual Studio Code**. O planejamento da interface deve considerar os **requisitos de software** (funcionalidades) e a navegação para criar a melhor experiência de usuário. |  |
| **Componentização** | Elementos que se repetem em mais de uma tela devem ser **"componentizados"** para serem reutilizados ao longo da aplicação. |  |
| **Container Principal** | O componente **<View>** deve ser usado como **container** de outros componentes, inclusive de outras Views, para auxiliar na organização do código e na estilização. |  |
| **Rolagem (Scrolling)** | Para exibir muito texto na tela, utiliza-se o **<ScrollView>**. No entanto, para listas grandes ou de tamanho desconhecido, componentes de lista otimizados (como o FlatList) são preferíveis. |  |
| **Componentes de Interação** | O **<Modal>** é uma estratégia para apresentar conteúdo adicional sem trocar de página ou aumentar o *scroll* vertical. Para botões, o **<Pressable>** é a implementação preferida, conforme a documentação oficial do React Native. |  |
| **Feedback ao Usuário** | O **<ActivityIndicator>** exibe o elemento visual de "loading" e é essencial para informar o usuário de que uma tarefa está sendo processada, especialmente durante requisições externas. |  |

--------------------------------------------------------------------------------

Módulo 2: Componentes de Lista e Multivalorados

Este módulo trata dos componentes otimizados para exibir coleções de dados.

|  |  |  |
| --- | --- | --- |
| Ponto Importante | Detalhe | Citação |
| **Tipos de Listas Nativas** | O React Native possui três principais componentes nativos de listas: **VirtualizedList, SectionList e FlatList**. |  |
| **VirtualizedList** | É a **implementação-base** dos outros dois componentes. Ele é otimizado para o consumo de memória e desempenho ao manusear **grandes listas**, utilizando a **manutenção de janela finita** para renderizar itens incrementalmente. |  |
| **SectionList** | Componente específico para a exibição de dados em **formato de lista seccionada** (agrupada), permitindo a inclusão de um título identificador em cada seção (ex: produtos por categoria). A propriedade **sections** é obrigatória. |  |
| **FlatList** | Otimizado para renderizar **listas simples e básicas**. Possui recursos como suporte a múltiplas plataformas, exibição horizontal, cabeçalho, rodapé e **separadores**. |  |
| **Recursos de Atualização** | O FlatList (e os demais) utiliza o **extraData** para monitorar e identificar atualizações nos dados. A função **onEndReached** é usada para carregar novos conteúdos (ex: *infinite scrolling*) quando o usuário chega ao final da lista. |  |

--------------------------------------------------------------------------------

Módulo 3: Principais Modelos de Navegação

O módulo descreve os três modelos fundamentais de navegação entre as telas (*screens*) de um aplicativo, geralmente implementados por meio de bibliotecas externas.

|  |  |  |
| --- | --- | --- |
| Ponto Importante | Detalhe | Citação |
| **Stack Navigation** | O modelo consiste no **empilhamento de telas** (como uma pilha LIFO), onde cada nova tela acessada é colocada por cima da anterior. É similar à navegação em um *site* web, pois provê **transição entre telas** e **gestão do histórico**. |  |
| **Tab Navigation** | Navegação composta por **guias ou abas**, geralmente inseridas na parte inferior do aplicativo. Usa createBottomTabNavigator. A navegação é feita ao pressionar os botões contidos na guia, em vez de usar um botão explícito em tela. |  |
| **Drawer Navigation** | Fornece um **menu de navegação inicialmente invisível**, que é acessado ao arrastar a tela da esquerda para o meio (ou pelo menu hambúrguer). Usa createDrawerNavigator. |  |
| **Combinação de Modelos** | Os modelos podem ser usados individualmente ou **combinados**, mas a aplicação deve ter **apenas um NavigationContainer**. A combinação é feita definindo um modelo como item (component) do outro. |  |

--------------------------------------------------------------------------------

Módulo 4: Recursos de Estilização e Animação

Este módulo trata de como aplicar aparência e movimento aos componentes do React Native.

|  |  |  |
| --- | --- | --- |
| Ponto Importante | Detalhe | Citação |
| **Estilização Nativa (style)** | A estilização utiliza um **código JavaScript** por meio da propriedade style. As propriedades são similares ao CSS, mas devem ser declaradas no padrão **camelCase** (ex: margin-top vira marginTop). |  |
| **Definição de Estilos** | Os estilos podem ser definidos de forma **inline**, **interna** (usando StyleSheet.create no mesmo script), ou **externa** (importando um script de estilo separado). |  |
| **Styled Components** | É uma biblioteca alternativa para estilizar componentes que permite o uso de **propriedades CSS tradicionais** de forma similar à web. |  |
| **Animated API** | Recurso nativo baseado em relações declarativas (entradas e saídas) para **animações controladas**. Permite animar componentes como View, Text, Image, e listas. É usada para efeitos como *fade in* ou *fade out* (ex: Animated.timing). |  |
| **LayoutAnimation API** | Recurso nativo voltado para a **animação do layout** do aplicativo durante o seu ciclo de renderização e atualização. É útil quando o redimensionamento de um elemento afeta seus vizinhos. Oferece **menos controle** do que a Animated API. |  |
| **Importância da Animação** | Animações melhoram a experiência do usuário, fornecendo **dicas visuais** sobre o que está acontecendo no aplicativo, sendo úteis em mudanças de estado (como carregamento de novo conteúdo) |  |

Exercícios:

Interface gráfica do usuário, Texto, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Word, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.