@fpftech.educacional

FPFtech
Escola Tecnológica

Coleções e Tratamento de Dados - Jetpack Compose

Objetivo: Trabalhar com listas (List, filter, map, sumOf) e exibir dados dinamicamente com @**Composables**.

Coleções em Kotlin são poderosas e concisas, ideais para trabalhar com listas de dados.

listOf → cria uma lista de itens

```
val numeros = listOf(1, 2, 3, 4, 5)
```

filter → filtra elementos com base em uma condição

```
val pares = numeros.filter { it % 2 == 0 } // [2, 4]
```

It = item, palavra reservada do Kotlin.

Coleções e Tratamento de Dados - Jetpack Compose

map → transforma cada item

```
val dobro = numeros.map { it * 2 } // [2, 4, 6, 8, 10]
```

sortedBy → ordena com base em algo

```
val nomes = listOf("Carlos", "Ana", "Beatriz")
val ordenado = nomes.sortedBy { it } // [Ana, Beatriz, Carlos]
```

sumOf → soma com base em uma propriedade

```
data class Produto(val nome: String, val preco: Double)
val produtos = listOf(Produto("Lápis", 1.5), Produto("Caderno", 10.0))
val total = produtos.sumOf { it.preco } // 11.5
```

data class – Simples e poderosa

```
data class Produto(val nome: String, val preco: Double)
```

Vantagens automáticas:

toString() → representação em texto

equals() → compara valores

copy() → cria uma cópia com alterações

```
val p1 = Produto("Caneta", 2.0)
val p2 = p1.copy(preco = 2.5)
println(p1 == p2) // false
```

- ideal para representar modelos de dados simples e reutilizáveis.

Exibição de Listas com Compose

lista com forEach:

```
@Composable
fun ListaSimples() {
    val nomes = listOf("Ana", "Bruno", "Clara")
    Column {
        nomes.forEach { nome ->
            Text("Nome: $nome")
        }
    }
}
```

Para listas maiores:

use LazyColumn

só renderiza o que está visível,

mais eficiente!

```
@Composable
fun ListaDeProdutos(produtos: List<Produto>) {
    LazyColumn {
        items(produtos.size) { index ->
            val produto = produtos[index]
            Text("${produto.nome} - R$${produto.preco}")
        }
    }
}
```

Jetpack Compose - Componentes

Text: Exibe texto na tela.

```
@Composable
fun ExemploTexto() {
    Text(text = "Olá, Jetpack Compose!", fontSize = 20.sp)
}
```

Button: clicável executa ação(click).

```
@Composable
fun ExemploBotao() {
    Button(onClick = { /* Ação a ser executada */ }) {
        Text(text = "Clique aqui")
    }
}
```

Image: Exibe imagem.

```
@Composable
fun ExemploImagem() {
    Image(
        painter = painterResource(id = R.drawable.exemplo_imagem),
        contentDescription = "Descrição da imagem",
        modifier = Modifier.size(128.dp)
    )
}
```

Jetpack Compose - Componentes

TextField:

Campo texto entrada do Usuário.

Remember: quando se quer restartar o component, ele lembra do estado anterior

```
@Composable
fun ExemploCampoTexto() {
    var texto by remember { mutableStateOf("") }
    TextField(
        value = texto,
        onValueChange = { texto = it },
        label = { Text("Digite algo") }
    )
}
```

Card: Agrupa co

relacionado.

```
@Composable
fun ExemploCard() {
    Card(
        modifier = Modifier.padding(16.dp),
        elevation = 8.dp
    ) {
        Column(modifier = Modifier.padding(16.dp)) {
            Text(text = "Título do Card", style = MaterialTheme.typography.h6)
            Text(text = "Conteúdo do card.")
        }
    }
}
```

Jetpack Compose - Componentes

Row: Organiza horizontal.

```
@Composable
fun ExemploRow() {
    Row(
        modifier = Modifier.fillMaxWidth(),
        horizontalArrangement = Arrangement.SpaceBetween
    ) {
        Text(text = "Esquerda")
        Text(text = "Direita")
    }
}
```

Column: Organiza vertical.

```
@Composable
fun ExemploColumn() {
    Column(
        modifier = Modifier.fillMaxHeight(),
        verticalArrangement = Arrangement.Center
    ) {
        Text(text = "Topo")
        Text(text = "Centro")
        Text(text = "Base")
    }
}
```

Jetpack Compose - Componentes

LazyColumn / LazyRow:

Carregam elementos somente quando necessário.

```
@Composable
fun ExemploLazyColumn() {
    val itens = listOf("Item 1", "Item 2", "Item 3", "Item 4")
    LazyColumn {
        items(itens) { item ->
            Text(text = item, modifier = Modifier.padding(8.dp))
        }
    }
}
```

Scaffold: Estrutura básica

Telas, fornece slots para componentes: *TopAppBar, BottomNavigation, FloatingActionButon, etc.*

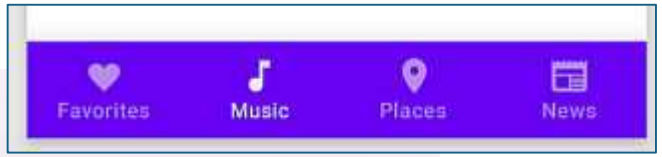
```
@Composable
fun ExemploScaffold() {
    Scaffold(
        topBar = {
            TopAppBar(title = { Text("Titulo da AppBar") })
        },
        floatingActionButton = {
            FloatingActionButton(onClick = { /* Ação */ }) {
                Icon(Icons.Default.Add, contentDescription = "Adicionar")
            }
        }
    ) { innerPadding ->
        // Conteúdo principal da tela
        Text(
            text = "Conteúdo da tela",
            modifier = Modifier.padding(innerPadding).padding(16.dp)
        )
    }
}
```

Jetpack Compose - Componentes

TopAppBar: Barra app no topo da tela.


```
@Composable
fun ExemploTopAppBar() {
    TopAppBar(
        title = { Text("Minha AppBar") },
        navigationIcon = {
            IconButton(onClick = { /* Navegar */ }) {
                Icon(Icons.Default.Menu, contentDescription = "Menu")
            }
        },
        actions = {
            IconButton(onClick = { /* Buscar */ }) {
                Icon(Icons.Default.Search, contentDescription = "Buscar")
            }
        }
    )
}
```

BottomNavigation: Barra navegação parte inferior tela.




```
@Composable
fun ExemploBottomNavigation() {
    val itens = listOf("Início", "Pesquisar", "Perfil")
    var selecionado by remember { mutableStateOf(0) }

    BottomNavigation {
        itens.forEachIndexed { index, item ->
            BottomNavigationItem(
                icon = { Icon(Icons.Default.Home, contentDescription = item) },
                label = { Text(item) },
                selected = selecionado == index,
                onClick = { selecionado = index }
            )
        }
    }
}
```

Jetpack Compose - Componentes

Snackbar: Notificaç

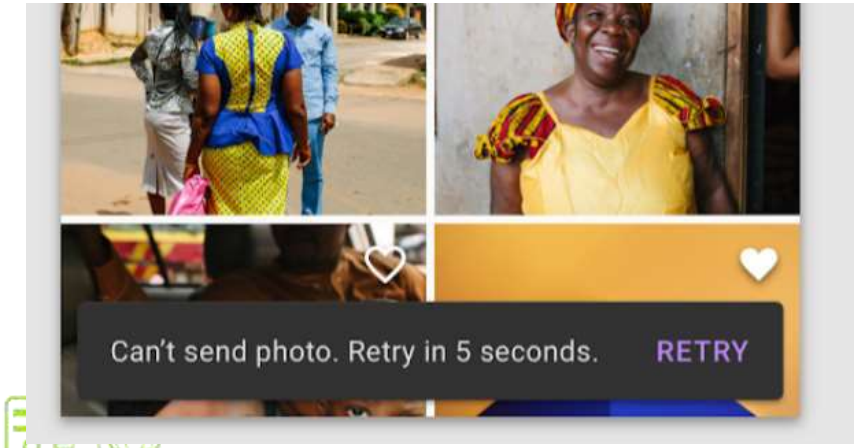
transitória que apare

na parte inferior da t

```
@Composable
fun ExemploSnackbar() {
    val snackbarHostState = remember { SnackbarHostState() }

    LaunchedEffect(Unit) {
        snackbarHostState.showSnackbar("Mensagem de aviso", actionLabel = "Des
    }

    Scaffold(
        snackbarHost = { SnackbarHost(hostState = snackbarHostState) }
    ) { innerPadding ->
        // Conteúdo da tela
    }
}
```



Mão na Massa

Lista:

```
package com.example.test

import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.material3.MaterialTheme
import androidx.compose.ui.Modifier
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
```

Mão na M

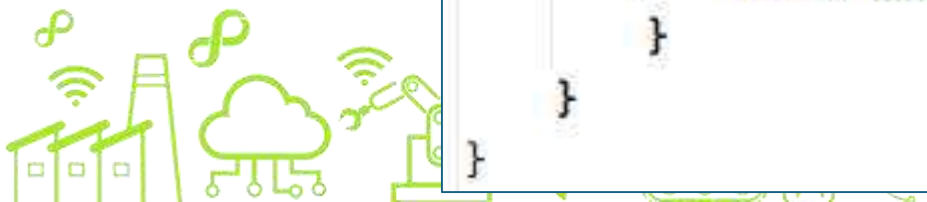
Lista:

```
@Composable
fun ListaDeProdutos(produtos: List<Produto>) {
    Column(modifier = Modifier.padding(16.dp)) {
        Text("Produtos em promoção:", style = MaterialTheme.typography.titleMedium)
        produtos
            .filter { it.emPromocao }
            .forEach {
                Text("- ${it.nome} por R$%.2f".format(it.preco))
            }

        Spacer(modifier = Modifier.height(8.dp))

        val total = produtos.sumOf { it.preco }
        val media = produtos.filter { it.emPromocao }.map { it.preco }.average()
        val maisCaro = produtos.filter { it.emPromocao }.maxByOrNull { it.preco }

        Text("Valor total do carrinho: R$%.2f".format(total), fontWeight = FontWeight.Bold)
        Text("Valor médio dos produtos: R$%.2f".format(media))
        maisCaro?.let {
            Text("Produto mais caro: ${it.nome} (R$%.2f)".format(it.preco))
        }
    }
}
```



Mão na Massa

Criar data class Produto(nome, preco, emPromocao)

Criar lista fixa de produtos

Exibir produtos em promoção

Exibir valor médio average() e produto mais caro usando maxByOrNull

Calcular e exibir valor total do carrinho (MAIN)

```
@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
    val produtos = listOf(
        Produto("Camiseta", 59.90, true),
        Produto("Tênis", 199.90, false),
        Produto("Boné", 39.90, true)
    )
    ListaDeProdutos(produtos)
}
```

FPFtech
Escola Tecnológica

Obrigado!

FPFtech
Escola Tecnológica

@fpftech.educacional