

## Estrutura do Código

O código é um aplicativo Android que utiliza Jetpack Compose para construir uma interface de usuário. A estrutura é organizada em uma classe principal (MainActivity) e funções composables que definem a UI.

### 1. Pacote e Importações

```
package com.example.appcadastro

import android.os.Bundle
```

**package com.example.appcadastro:** Define o namespace do aplicativo. É uma prática comum em projetos Java/Kotlin para organizar o código.

**Importações:** Inclui classes e funções necessárias para a construção da interface, manipulação de estado, e temas. O uso de `androidx.compose` indica que estamos utilizando a biblioteca Jetpack Compose.

### 2. Classe MainActivity

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            AppCadastroTheme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                    appPreview()
                }
            }
        }
    }
}
```

**class MainActivity : ComponentActivity():** Define a classe principal da atividade. `ComponentActivity` é uma classe base que permite o uso de Jetpack Compose.

**override fun onCreate(savedInstanceState: Bundle?):** Método do ciclo de vida da atividade que é chamado quando a atividade é criada. Aqui, inicializamos a interface do usuário.

**super.onCreate(savedInstanceState):** Chama o método da superclasse para garantir que a atividade seja inicializada corretamente.

**enableEdgeToEdge():** Permite que a interface utilize toda a tela, sem margens. Isso é útil para uma experiência visual mais imersiva.

**setContent { ... }:** Define o conteúdo da atividade usando uma função composable. O bloco dentro de `setContent` é onde a UI é construída.

**AppCadastroTheme { ... }:** Aplica um tema ao aplicativo. O tema pode definir cores, tipografia e outros estilos.

**Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding -> ... }**: Scaffold é um layout que fornece uma estrutura básica para a interface, incluindo áreas para o conteúdo principal, barras de navegação, etc. O modificador fillMaxSize() faz com que o Scaffold ocupe toda a tela.

### 3. Função Composable ProdutoItem

```
@Composable
fun ProdutoItem(){
    var nome by remember { mutableStateOf( value: "" ) }
    var telefone by remember { mutableStateOf( value: "" )}
    var curso by remember { mutableStateOf( value: "" ) }
    var serie by remember { mutableStateOf( value: "" ) }
    var observacao by remember { mutableStateOf( value: "" ) }
```

**@Composable**: Anotação que indica que a função pode ser usada para construir a interface de usuário. Composables são funções que podem ser chamadas para renderizar UI.

**var nome by remember { mutableStateOf("") }**: Cria uma variável nome que é observável. O uso de remember garante que o estado persista entre recomposições. mutableStateOf("") inicializa a variável com uma string vazia.

As outras variáveis (telefone, curso, serie, observacao) são criadas da mesma forma, cada uma representando um campo de entrada do usuário.

### Layout da Interface

```
Column(
    modifier = Modifier
        .fillMaxSize()
        .verticalScroll(rememberScrollState()),
    horizontalAlignment = Alignment.CenterHorizontally
){
```

**Column**: Um layout que organiza seus filhos verticalmente. É útil para empilhar elementos um em cima do outro.

**modifier = Modifier.fillMaxSize()**: O modificador fillMaxSize() faz com que a coluna ocupe todo o espaço disponível.

**verticalScroll(rememberScrollState())**: Permite que a coluna seja rolável verticalmente. rememberScrollState() cria um estado que mantém a posição de rolagem.

**horizontalAlignment = Alignment.CenterHorizontally**: Alinha todos os elementos filhos horizontalmente ao centro.

## Cabeçalho e Imagem

```
Box(  
    modifier = Modifier  
        .height(180.dp)  
        .background(  
            brush = Brush.horizontalGradient(  
                colors = listOf(  
                    Darkblue, Lightblue  
                )  
            )  
        )  
        .fillMaxWidth()  
)  
{  
    Image(  
        painter = painterResource(R.drawable.avatr_woman),  
        contentDescription = "avatar image",  
        contentScale = ContentScale.Crop,  
        modifier = Modifier  
            .offset(y = 70.dp)  
            .size(150.dp)  
            .clip(shape = CircleShape)  
            .align(Alignment.Center)  
    )  
}
```

**Box:** Um layout que permite sobrepor elementos. Aqui, é usado para criar um cabeçalho com uma imagem de avatar.

**modifier = Modifier.height(180.dp):** Define a altura do Box como 180 dp (density-independent pixels).

**background(brush = Brush.horizontalGradient(...)):** Aplica um fundo com um gradiente horizontal. Brush.horizontalGradient cria um gradiente entre as cores especificadas.

**fillMaxWidth():** Faz com que o Box ocupe toda a largura disponível.

**Image(...):** Renderiza uma imagem de recurso.

**painter = painterResource(R.drawable.avatr\_woman):** Carrega a imagem do recurso drawable.

**contentDescription = "avatar image":** Fornece uma descrição da imagem para acessibilidade.

**contentScale = ContentScale.Crop:** Define como a imagem deve ser escalada dentro do espaço disponível. Crop significa que a imagem será cortada para preencher o espaço.

**modifier = Modifier.offset(y = 70.dp):** Move a imagem 70 dp para baixo em relação à sua posição original.

**size(150.dp):** Define a largura e altura da imagem como 150 dp.

**clip(shape = CircleShape):** Aplica uma máscara à imagem, neste caso, um formato circular (CircleShape).

**align(Alignment.Center):** Alinha a imagem ao centro do Box.

## Campos de Entrada

Os campos de entrada são criados usando TextField, que permite ao usuário inserir dados. Cada campo é precedido por um Text que serve como rótulo.

```
Text(  
    text = "Cadastre-se",  
    fontFamily = FontFamily.SansSerif,  
    fontWeight = FontWeight(weight: 500),  
    fontSize = 35.sp,  
    modifier = Modifier.align(Alignment.CenterHorizontally)  
)
```

**Text(...):** Renderiza um texto na tela.

**text = "Nome: ":** O texto a ser exibido.

**fontFamily = FontFamily.SansSerif:** Define a família da fonte como sans-serif.

**fontWeight = FontWeight(250):** Define o peso da fonte. Valores mais altos indicam uma fonte mais grossa.

**fontSize = 25.sp:** Define o tamanho da fonte como 25 sp (scale-independent pixels), que é escalável com as configurações de acessibilidade do usuário.

**modifier = Modifier.align(Alignment.Start):** Alinha o texto ao início (lado esquerdo) do layout.

```
TextField(  
    value = nome,  
    onChange = { nome = it },  
    label = { Text(text: "Digite seu nome completo") },  
    maxLines = 2,  
    modifier = Modifier.fillMaxWidth(),  
)
```

**TextField(...):** Um componente de entrada de texto.

**value = nome:** O valor atual do campo de texto, que é ligado ao estado nome.

**onChange = { nome = it }:** Um lambda que é chamado sempre que o texto é alterado. it representa o novo valor do campo de texto, que é atribuído à variável nome.

**label = { Text("Digite seu nome completo") }:** Um rótulo que aparece acima do campo de texto. É uma função composável que renderiza o texto do rótulo.

**maxLines = 2:** Define o número máximo de linhas que o campo de texto pode exibir. Isso é útil para campos que podem ter múltiplas linhas de entrada, como um nome completo.

**modifier = Modifier.fillMaxWidth():** Faz com que o campo de texto ocupe toda a largura disponível.

**keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Number):** Para campos que devem aceitar apenas números (como telefone e série), este parâmetro é adicionado para especificar que o teclado numérico deve ser exibido.

## Botão de Cadastro

```
Button(  
    onClick = {  
        nome = ""  
        telefone = ""  
        curso = ""  
        serie = ""  
        observacao = ""  
    },  
    modifier = Modifier.align(Alignment.CenterHorizontally)  
) {  
    Text(text = "Cadastrar")  
}
```

**Button(...):** Um componente interativo que executa uma ação quando clicado.

**onClick = { ... }:** Um lambda que define a ação a ser executada quando o botão é clicado. Neste caso, ele limpa todos os campos de entrada, redefinindo as variáveis para strings vazias.

**modifier = Modifier.align(Alignment.CenterHorizontally):** Alinha o botão ao centro horizontalmente.

**Text(text = "Cadastrar"):** O texto exibido dentro do botão.

#### 4. Função de Preview appPreview

```
@Preview
@Composable
fun appPreview() {
    AppCadastroTheme {
        Surface(
            modifier = Modifier.fillMaxSize(),
            color = MaterialTheme.colorScheme.background
        ) {
            ProdutoItem()
        }
    }
}
```

**@Preview:** Anotação que permite visualizar a interface no Android Studio sem a necessidade de executar o aplicativo. Isso é útil para desenvolvimento e design de UI.

**AppCadastroTheme { ... }:** Aplica o tema ao preview, garantindo que a visualização reflita o estilo do aplicativo.

**Surface(...):** Um contêiner que aplica um fundo e outros estilos.

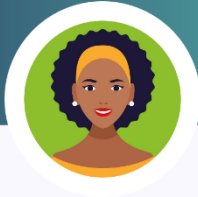
**modifier = Modifier.fillMaxSize():** Faz com que a superfície ocupe todo o espaço disponível.

**color = MaterialTheme.colorScheme.background:** Define a cor de fundo da superfície com base no tema atual.

**ProdutoItem():** Chama a função composável que renderiza a interface de cadastro.

O código implementa um aplicativo de cadastro utilizando Jetpack Compose, que é uma abordagem moderna e declarativa para construir interfaces de usuário no Android. Cada componente e parâmetro foi projetado para criar uma experiência de usuário fluida e reativa. O uso de estados mutáveis, layouts responsivos e componentes interativos permite que o aplicativo responda às ações do usuário de forma eficiente.

1:58



## Cadastre-se

Nome:

Digite seu nome completo

Telefone:

Digite o telefone...

Curso:

Digite o nome do curso

Série:

Digite o número da sua série

Observação:

Digite suas observações

Cadastrar

2:00



## Cadastre-se

Nome:

Digite seu nome completo  
**Wallex**

Telefone:

Digite o telefone...  
**11956488217**

Curso:

Digite o nome do curso  
**Análise e Desenvolvimento de Sistemas**

Série:

Digite o número da sua série  
**2**

Observação:

Digite suas observações  
**Bom app**

Cadastrar