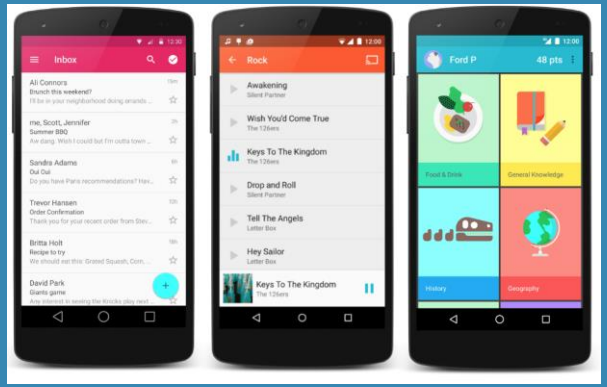
soluções mobile

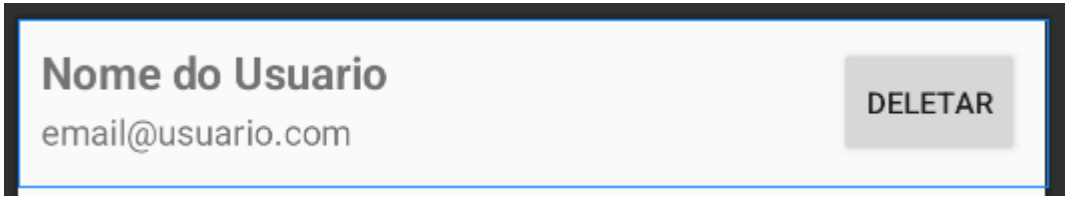
prof. Thyerri Mezzari

Android: Criar listas dinâmicas com o RecyclerView

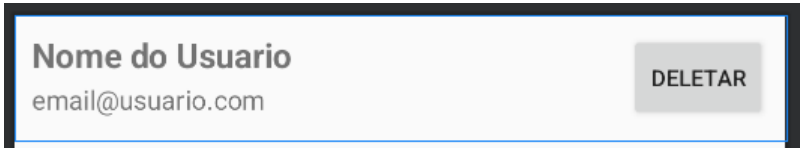


RecyclerView item

Crie um layout de como será um item em nossa lista. Use o que for preciso para chegar ao resultado necessário. Linear layout, Relative Layout, Buttons, TextViews, etc.



RecyclerView item




```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="9dp">

    <LinearLayout
        android:orientation="vertical"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="wrap_content">

        <TextView
            android:id="@+id/nome"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="2dp"
            android:text="Nome do Usuario"
            android:textSize="18sp"
            android:textStyle="bold" />

        <TextView
            android:id="@+id/email"
            android:text="email@usuario.com"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            />

    </LinearLayout>

    <Button
        android:id="@+id/btDelete"
        style="@style/Widget.AppCompat.Button.Small"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Deletar"
        android:textSize="12sp" />

</LinearLayout>
```

Layout XML

Insira o RecyclerView no arquivo XML onde você deseja exibir a lista. Ajuste largura e altura conforme necessidade de sua tela.

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerview"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:listitem="@layout/recyclerview_item"
    android:layout_marginTop="22dp" />
```

Layout XML

Insira o RecyclerView no arquivo XML onde você deseja exibir a lista. Ajuste largura e altura conforme necessidade de sua tela.

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerview"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:listitem="@layout/recyclerview_item"
    android:layout_marginTop="22dp" />
```

List Adapter

Para “gerenciar” os dados que serão exibidos nesta lista, precisamos de uma classe que seja filha de ListAdapter e implemente um ViewHolder.

```
class CustomAdapter(private val dataSet: Array<String>) :
    RecyclerView.Adapter<CustomAdapter.ViewHolder>() {

    /**
     * Provide a reference to the type of views that you are using
     * (custom ViewHolder)
     */
    class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val textView: TextView

        init {
            // Define click listener for the ViewHolder's View
            textView = view.findViewById(R.id.textView)
        }
    }

    // Create new views (invoked by the layout manager)
    override fun onCreateViewHolder(viewGroup: ViewGroup, viewType: Int):
ViewHolder {
        // Create a new view, which defines the UI of the list item
        val view = LayoutInflater.from(viewGroup.context)
            .inflate(R.layout.text_row_item, viewGroup, false)

        return ViewHolder(view)
    }

    // Replace the contents of a view (invoked by the layout manager)
    override fun onBindViewHolder(viewHolder: ViewHolder, position: Int) {

        // Get element from your dataset at this position and replace the
        // contents of the view with that element
        viewHolder.textView.text = dataSet[position]
    }

    // Return the size of your dataset (invoked by the layout manager)
    override fun getItemCount() = dataSet.size

}
```

List Adapter

O adapter pode parecer confuso e difícil. Crie seu adapter conforme o exemplo no slide anterior seguindo as dicas:

- Lembre-se que você sempre pode usar o atalho ALT + ENTER para resolver problemas no Android Studio como importar dependências (imports) ou inserir trechos de códigos que devem ser implementados.
- Na dúvida consulte a documentação oficial em developer.android.com.
- Para recycler view o passo a passo está em https://developer.android.com/develop/ui/views/layout/recyclerview?hl=pt-br#kotlin

List Adapter

O adapter pode parecer confuso e difícil. Crie seu adapter conforme o exemplo no slide anterior seguindo as dicas:

- Lembre-se do conceito principal da recycler view que é reaproveitar os recursos em tela, apenas substituindo os dados, fazendo assim economizar processamento e memória.
- Leia os métodos e suas documentações.

List Adapter - View Holder

Precisamos implementar corretamente os métodos de nosso Adapter e View Holder.

init (construtor(

onBindViewHolder;

onCreateViewHolder

getItemCount

List Adapter - View Holder

No método init {} de nosso ViewHolder devemos pegar a referência dos nossos componentes no layout do item (recyclerViewItem).

Faremos isso com findViewById, assim como fazemos em uma Activity.

```
class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {
    val textView: TextView

    init {
        // Define click listener for the ViewHolder's View
        textView = view.findViewById(R.id.textView)
    }
}
```

List Adapter - View Holder

No método onCreateViewHolder, precisamos “inflar” o layout do nosso recyclerViewItem e retornar um ViewHolder passando o objeto de View.

```
// Create new views (invoked by the layout manager)
override fun onCreateViewHolder(viewGroup: ViewGroup, viewType: Int):
ViewHolder {
    // Create a new view, which defines the UI of the list item
    val view = LayoutInflater.from(viewGroup.context)
        .inflate(R.layout.text_row_item, viewGroup, false)

    return ViewHolder(view)
}
```

List Adapter - View Holder

No método onBindViewHolder, é onde vamos “montar” os dados em nosso recyclerViewItem, preenchendo as views com a informação real do item.

```
// Replace the contents of a view (invoked by the layout manager)
override fun onBindViewHolder(viewHolder: ViewHolder, position: Int) {

    // Get element from your dataset at this position and replace the
    // contents of the view with that element
    viewHolder.textView.text = dataSet[position]
}
```

List Adapter - View Holder

No método getItemCount, devemos retornar a quantidade de itens que nosso data set (lista) possui. Assim nosso adapter sempre saberá como lidar com os itens de nossa lista.

```
// Return the size of your dataset (invoked by the layout manager)
override fun getItemCount() = dataSet.size
```

RecyclerView

Agora devemos instanciar nossa RV na Activity ou Fragment onde ela deverá aparecer.

Utilize R.id. recycler_view com o ID correto que você referenciou no arquivo XML.

```
val dataset = arrayOf("January", "February", "March")
val customAdapter = CustomAdapter(dataset)

val recyclerView: RecyclerView = findViewById(R.id.recycler_view)
recyclerView.adapter = customAdapter
recyclerView.layoutManager = LinearLayoutManager(this)
```

RecyclerView - Atualizar dados

Para atualizar os dados na tela é possível add itens em nosso objeto List (ou remover) e chamar adapter.notifyDataSetChanged().

Alternativamente pode ser usado mAdapter.notifyItemInserted(mItems.size() - 1) para ter um desempenho melhor.

Ou, caso atualize a lista inteira. Chame adapter.submitList(minhaLista: List)

obrigado 🚀