DESENVOLVIMENTO MOBILE COM ANDROID

André Guedes
Robson Moreira

Framework, Android Studio e SDK Tools

Introdução

- Android é um sistema operacional para dispositivos móveis criado a partir do core do Linux;
- Adquirido pela Google, em 2005, através da compra da empresa Android Inc., fundada por Andy Rubinera, Nick Sears e Chris White;
- Anunciado em 5 de novembro de 2007, junto com a criação da *Open Handset Alliance* (OHA).

Por que Android?

- A cada dia mais de 1 milhão de novos dispositivos são ativados;
- 1,5 bilhões de aplicativos baixados por mês;
- Embarcado em smartphones, tablets, relógios, TVs e automóveis;
- Presente em 86,8% dos smartphones vendidos em 2016.

Participação de Mercado

Period	Android	iOS	Windows Phone	Others
2015Q4	79.6%	18.7%	1.2%	0.5%
2016Q1	83.5%	15.4%	0.8%	0.4%
2016Q2	87.6%	11.7%	0.4%	0.3%
2016Q3	86.8%	12.5%	0.3%	0.4%

Source: IDC, Nov 2016

Histórico das Versões

Cupcake
Android 1.5

Donut
Android 1.6

Eclair
Android 2.0/2.1

Froyo
Android 2.2.x

Gingerbread
Android 2.3.x

Honeycomb
Android 3.x

Ice Cream Sandwich
Android 4.0.x

Jelly Bean
Android 4.1.x

KitKat
Android 4.4.x

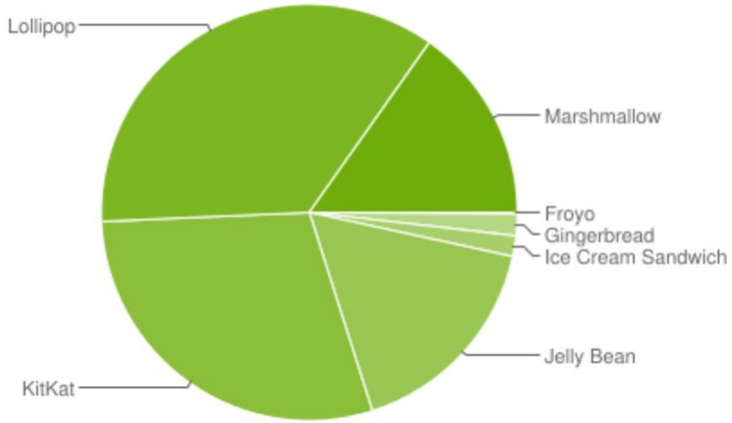
Lollipop
Android 5.0

Marshmallow
android 6.0

Nougat
android 7.0

Distribuição das Versões

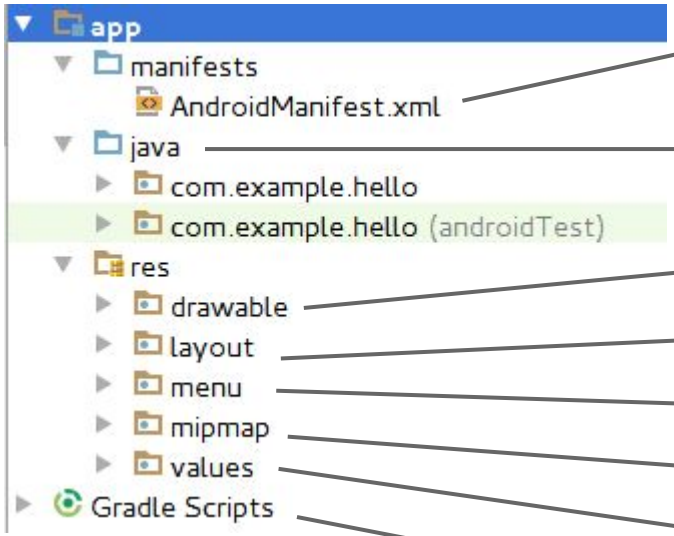
Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.6%
4.1.x	Jelly Bean	16	6.0%
4.2.x		17	8.3%
4.3		18	2.4%
4.4	KitKat	19	29.2%
5.0	Lollipop	21	14.1%
5.1		22	21.4%
6.0	Marshmallow	23	15.2%



Ambiente de Desenvolvimento

- JDK (Java Development Kit);
- SDK (Software Development Kit);
- AVD (Android Virtual Device);
- IDE (Integrated Development Environment);
  - Android Studio: build via Gradle;

Estrutura do projeto na visão Android



Configurações do App

Código fonte do App (**Code Guidelines**)

Imagens

Definições de tela

Definições de Menus

Ícones

Definição de Estilos e Strings (i18n)

Scripts de Build e Gestão de Dependências

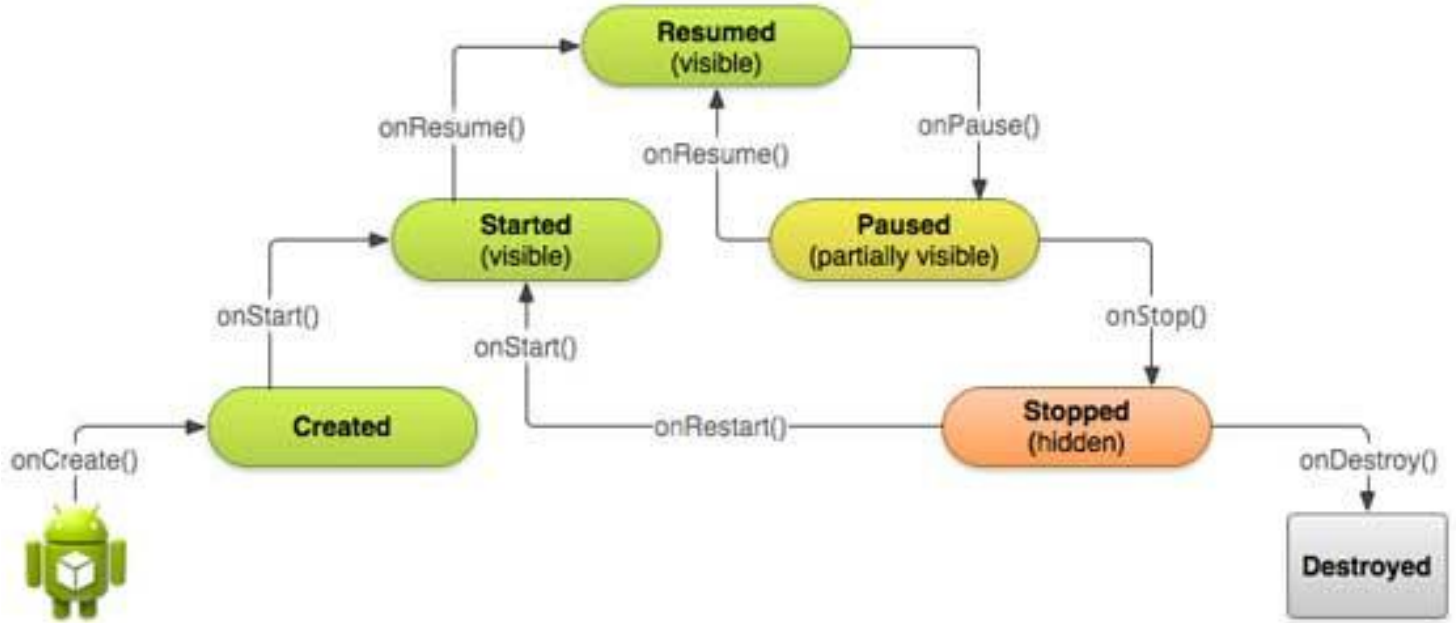

SHOW ME
{ the Code }

Activities, Intents e Layouts

Ciclo de vida de uma Activity

- **onCreate**: chamado quando a Activity é criada;
- **onStart**: chamado após o onCreate, e antes da Activity se tornar visível;
- **onResume**: chamado após o onStart, quando a Activity se torna visível;
- **onPause**: é chamado após o onResume, quando a Activity está para perder a visibilidade para outro componente;
- **onStop**: a Activity não está mais visível para o usuário;
- **onDestroy**: a Activity está prestes a ser destruída.

Ciclo de vida de uma Activity



Intents

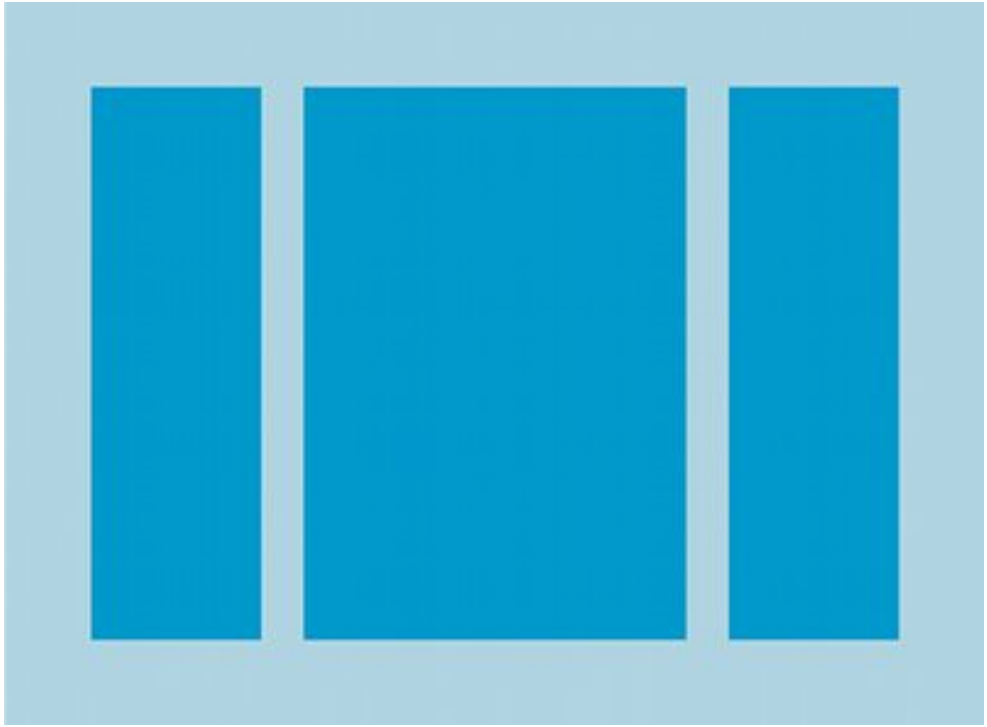
- Intents são objetos responsáveis por passar informações, como se fossem mensagens, para os principais componentes da API do Android, como as *Activities*, *Services* ou *BroadCast Receivers*:

```
Intent intent = new Intent(ActivityOrigem.this, ActivityDestino.class);
intent.putExtra(ActivityDestino.CHAVE, seuObjeto); // Opcional
startActivity(intent);
```

Linear Layout

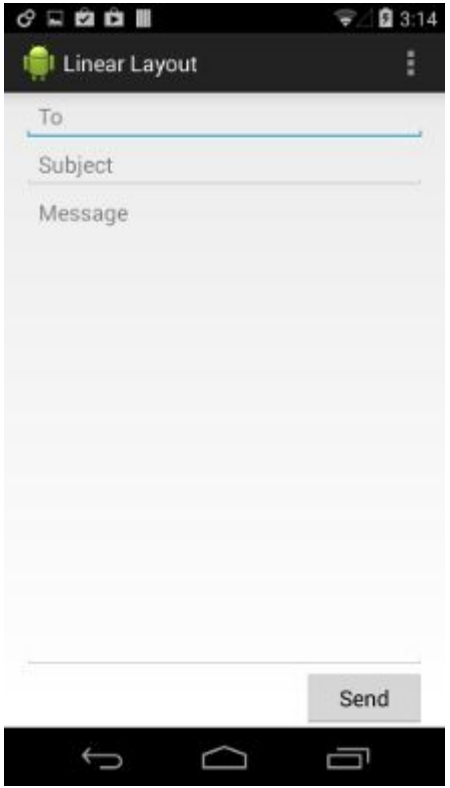
- Grupo de exibições que alinha todos os filhos em uma única direção. Vertical ou Horizontal;
- Seus filhos são empilhados um após o outro;
- Permite atribuir ponderações a filhos individuais, atribuindo sua importância em uma exibição em termos de quanto espaço ele deve ocupar na tela.

Linear Layout



Linear Layout

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:orientation="vertical" >
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/to" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/subject" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:gravity="top"
        android:hint="@string/message" />
    <Button
        android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_gravity="right"
        android:text="@string/send" />
</LinearLayout>
```



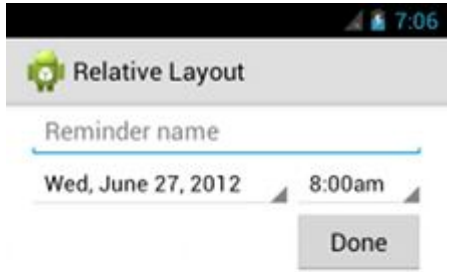
Relative Layout

- Grupo de exibições que alinha seus filhos em posições relativas;
- A posição de cada exibição pode ser especificada em relação aos seus irmãos (como a esquerda ou abaixo);
- Muito poderoso caso conheça todos os atributos.

Relative Layout

Relative Layout

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp" >
    <EditText
        android:id="@+id/name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/reminder" />
    <Spinner
        android:id="@+id/dates"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_below="@id/name"
        android:layout_alignParentLeft="true"
        android:layout_toLeftOf="@+id/times" />
    <Spinner
        android:id="@id/times"
        android:layout_width="96dp"
        android:layout_height="wrap_content"
        android:layout_below="@id/name"
        android:layout_alignParentRight="true" />
    <Button
        android:layout_width="96dp"
        android:layout_height="wrap_content"
        android:layout_below="@id/times"
        android:layout_alignParentRight="true"
        android:text="@string/done" />
</RelativeLayout>
```




SHOW ME
{ the Code }

ListView e RecyclerView

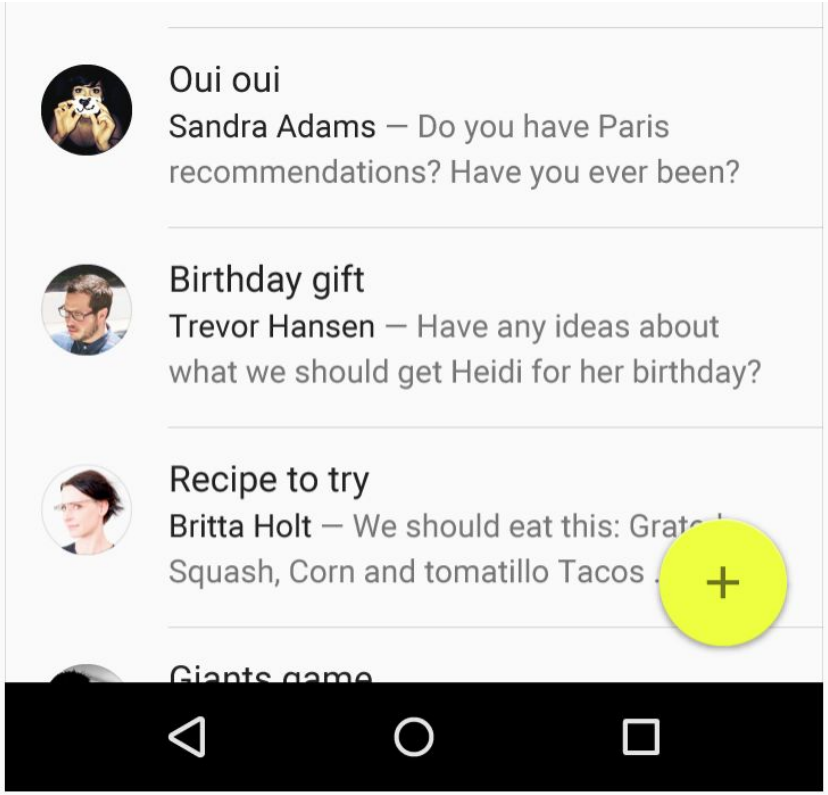
Listas no Android

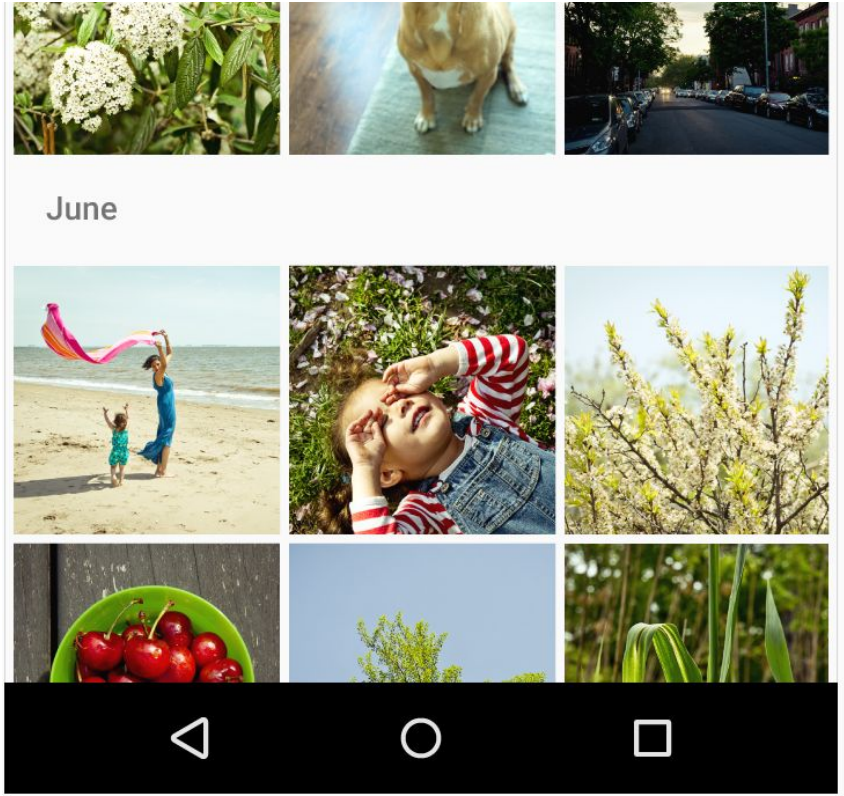
- ListView;
- GridView;
- RecyclerView.

ListView e GridView

- ListView - Grupo de exibições que exibe uma lista de itens roláveis;
- GridView - Grupo de exibições que exibe os itens em uma grade bidimensional rolável;
- ListView e GridView - Presente desde da primeira versão do Android.

Exemplos: ListView e GridView



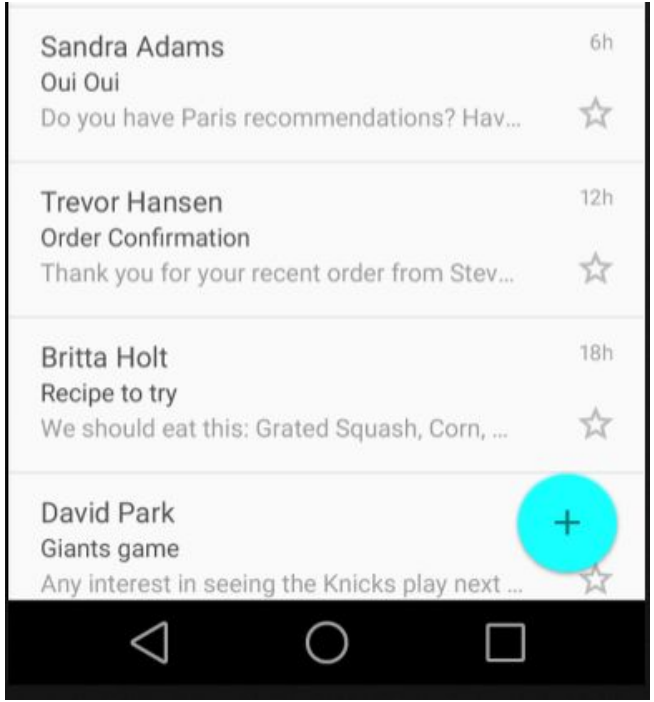


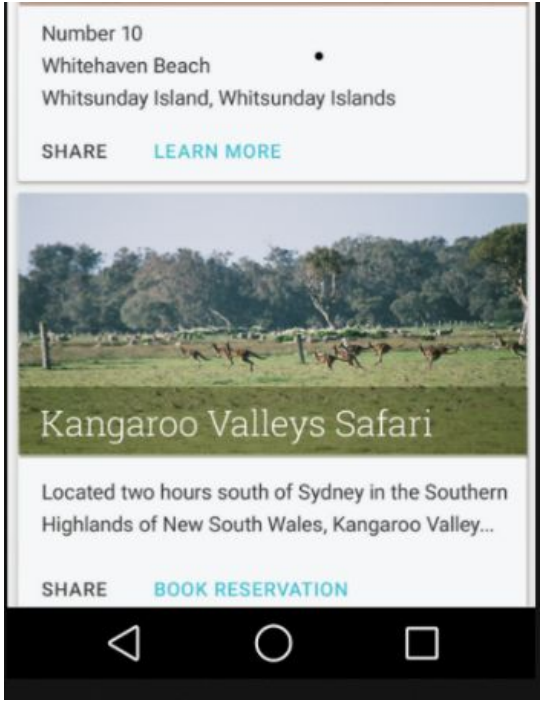
RecyclerView

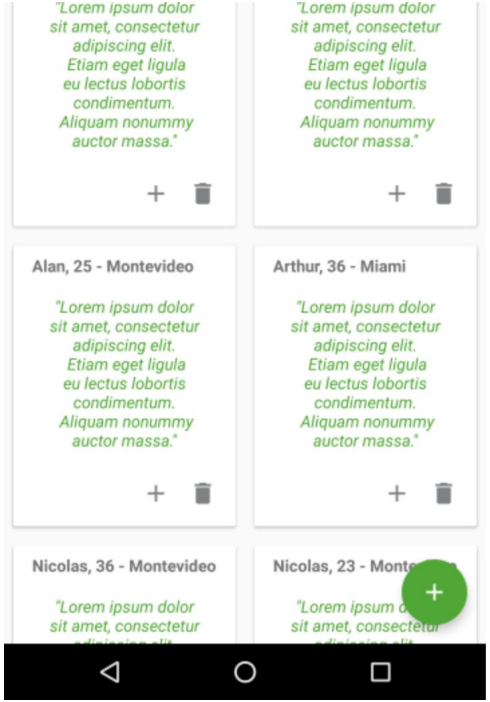
- Lista e cartões complexos com Material Design;
- Versão avançada e flexível do ListView e GridView;
- Simplifica a exibição e o tratamento de grande conjuntos de dados;
- Gerenciador de layout para posicionar itens.

```
compile 'com.android.support:recyclerview-v7:25.3.1'
compile 'com.android.support:cardview-v7:25.3.1'
compile 'com.android.support:design:25.3.1'
```

Exemplos: RecyclerView



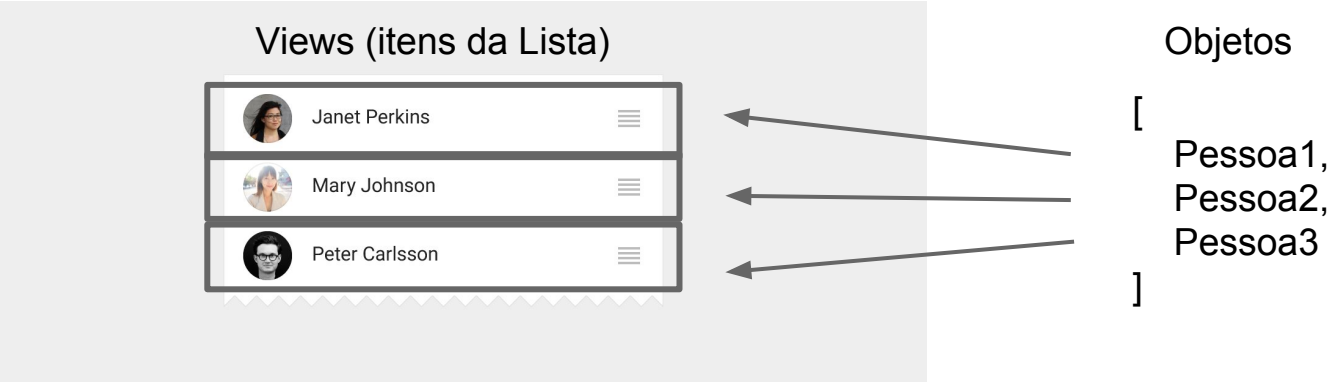




Componentes no RecyclerView

Adapters

- Funciona como uma ponte entre uma ou mais Views para apresentação dos elementos de uma fonte de dados:




SHOW ME
{ the Code }

Armazenamento no Android e RESTFul

Opções de armazenamento

- Preferencias compartilhadas (SharedPreferences);
- Armazenamento interno;
- Armazenamento externo;
- Banco de dados SQLite.

O que é Rest?

- Conjunto de princípios arquiteturais
- Protocolo HTTP
- Surgiu em 2000 por Roy Fielding (um dos principais autores da especificação do HTTP)

Requisições

- Comunicação através de mensagens
- Requisições “possuem” todo o conteúdo necessário para o processamento de sua resposta

Recursos

- URI (Identificador Uniforme de Recursos)

```
http://meudominio/urso
http://meudominio/urso/11
```

Métodos HTTP

- GET
- POST
- PUT
- DELETE
- HEAD
- OPTIONS

JSON

- JavaScript Object Notation
- Formato leve para intercâmbio de dados
- Amplamente utilizado em API’s REST

Exemplo:


```
{
    "_id" : "554b5510646268050f790623",
    "value" : 150.3,
    "client" : "Urso",
    "description" : "Levar mel para o urso",
    "paid" : true,
    "date" : "2015-01-01T00:00:00.000Z",
    "address" : "Canada"
}
```

AsyncTask

- Uso adequado da UI Thread;
- Permite executar operações em segundo plano;
- Publica o resultado na UI Thread;
- Criada para facilitar o processamento em background.

AsyncTask: Tipos genéricos

- Params;
- Progress;
- Result.

AsyncTask: Métodos

- onPreExecute();
- doInBackground();
- onProgressUpdate();
- onPostExecute();


SHOW ME
{ the Code }

Themes e Material Design

Material Design

- Estabelece padrões de ícones, cores, animações, tipografia e hierarquias.
- O uso da palavra “Material” não é casual: o novo padrão tenta criar experiências próximas das interações que temos com objetos “reais”.

Themes e Material Design

- Paleta de Cores;
- Personalização na Paleta de Cores;
- Botões;
- Tipografia.

Floating Labels

- Guideline;
- TextInputLayout;
- Biblioteca de suporte.

```
compile 'com.android.support:design:25.3.1'
```

TextInputLayout

```
<android.support.design.widget.TextInputLayout
    android:id="@+id/usernameWrapper"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <EditText
        android:id="@+id/username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textEmailAddress"
        android:hint="Username"/>

</android.support.design.widget.TextInputLayout>
```

Floating Button

- Guideline;
- FloatingActionButton;
- Biblioteca de suporte.

```
compile 'com.android.support:design:25.3.1'
```

FloatingActionButton

```
<android.support.design.widget.FloatingActionButton
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|right"
    android:layout_margin="16dp"
    android:src="@drawable/ic_done"
    app:layout_anchor="@id/lvToDoList"
    app:layout_anchorGravity="bottom|right|end" />
```


SHOW ME
{ the Code }

Link para os projetos

https://github.com/andreguedes/minicurso_uniara

DÚVIDAS?