

# Aula 04 – Objetos Gráficos

Programação em Java para a Plataforma Android



# Agenda

- Depuração simples: imprimindo eventos
- Lidando com botões e spinners
- Carregando múltiplas atividades em uma aplicação
- O padrão adapter
- Mostrando vários itens com listas
- Os diversos layouts



# Views (de novo)

- Visão (view): um objeto que “sabe” se desenhar na tela
- Grupos (ViewGroup): uma visão que pode conter outras
- Layout: uma visão que “sabe” dispor outras visões na tela do aparelho

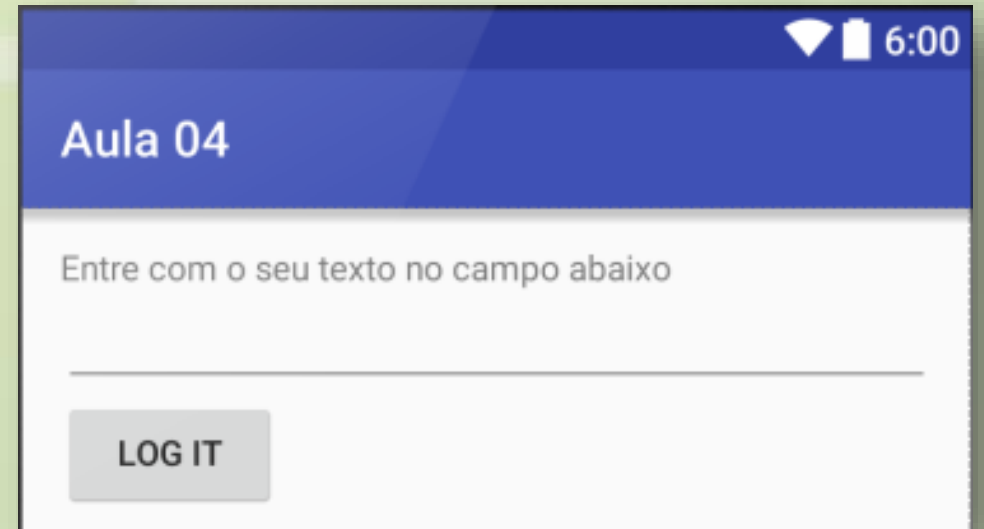


# Textos e Botões

## Primeira aplicação

Crie a visão mostrada ao lado

- Um clique no botão faz com que o texto exibido seja impresso no log do Android
- O log do Android é um arquivo de texto que pode ser usado para depuração



# Log do Android

Aula 04 - [C:\Users\Agripino\StudioProjects\Aula04] - [app] - ...app\src\main\java\com\cursoandroid\aula04\MainActivity.java - Android Studio 1.4

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Android

app

src

main

java

com

cursoandroid

aula04

MainActivity

activity\_main.xml

strings.xml

MainActivity.java

```
package com.cursoandroid.aula04;

import ...

public class MainActivity extends AppCompatActivity {

    private EditText editText;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Android Monitor

Emulator Nexus\_4\_API\_23 Android 6.0 (API 23)

com.cursoandroid.aula04 (23456)

Logcat

Memory

CPU

GPU

Network

Log level: Verbose

Regex

Show only selected application

03-27 20:37:55.200 22104-22147/com.cursoandroid.aula04 E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7f35ef099d70

03-27 20:37:55.880 2023-2296/com.google.android.gms I/Icing: Indexing 1A44626D05D6433B19F3B858B1E97A2EEC477BB9 from com.google.android.gms

03-27 20:37:56.020 23456-23456/com.cursoandroid.aula04 I/Choreographer: Skipped 54 frames! The application may be doing too much work on its main thread.

03-27 20:37:56.020 2023-2296/com.google.android.gms I/Icing: Not enough disk space for indexing trimmable

03-27 20:37:56.030 1362-1381/system\_process I/ActivityManager: Displayed com.cursoandroid.aula04.MainActivity: +1s660ms

03-27 20:37:56.060 1362-1736/system\_process I/WindowManager: Switching to real app window: Window{1ba0db2 u0 com.cursoandroid.aula04/com.cursoandroid.aula04.MainActivity}

03-27 20:37:56.110 1670-13981/com.android.inputmethod.latin E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7f35ef09b5f0

03-27 20:37:56.140 1456-1468/com.android.systemui W/art: Suspending all threads took: 10ms

03-27 20:37:56.140 1456-1468/com.android.systemui I/art: Background partial concurrent mark sweep GC freed 45323(1791KB) AllocSpace objects, 0(0B) LOS objects, 34% free, 7MB/11MB

03-27 20:37:56.240 1362-1985/system\_process W/art: Long monitor contention event with owner method=void com.android.server.am.ActivityManagerService.activityIdle(android.os.IBinder)

03-27 20:37:56.400 2023-2296/com.google.android.gms I/Icing: Not enough disk space for indexing trimmable

03-27 20:37:56.410 2023-2296/com.google.android.gms I/Icing: Indexing done 1A44626D05D6433B19F3B858B1E97A2EEC477BB9

03-27 20:37:56.410 2023-2296/com.google.android.gms I/Icing: Indexing BB748DD0679FA5C0BF35FEA5901EDAA07FBE7629 from com.google.android.googlequicksearchbox

03-27 20:37:56.410 2023-2296/com.google.android.gms I/Icing: Not enough disk space for indexing trimmable

03-27 20:37:56.490 2023-2296/com.google.android.gms I/Icing: Not enough disk space for indexing trimmable

03-27 20:37:56.500 2023-2296/com.google.android.gms I/Icing: Indexing done BB748DD0679FA5C0BF35FEA5901EDAA07FBE7629

03-27 20:38:51.440 23456-23456/com.cursoandroid.aula04 V/Aula04: Aula 04

Terminal

Android Monitor

Messages

Run

TODO

Session 'app': running (a minute ago)

4640:98 CRLF= UTF-8= Context: <no context>

## Saída crua

Podemos usar esse log para recolher informações de depuração e testes de desempenho



# Log do Android

- Texto é impresso pela classe Log, que possui cinco métodos estáticos. Cada método imprime em uma cor diferente:
  - Log.v: texto verboso (preto)
  - Log.d: texto de depuração (azul)
  - Log.i: texto informativo (verde)
  - Log.w: texto de aviso (laranja)
  - Log.e: texto de erro (vermelho)



# Código XML da view

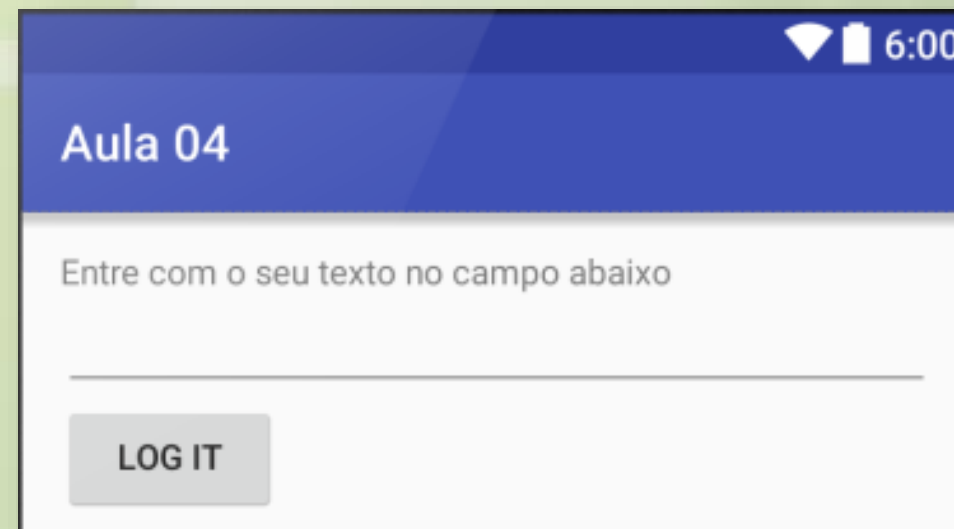
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView1"
        android:text="@string/entre_com_o_seu_texto_no_campo_abaixo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <EditText
        android:id="@+id/editText1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />

    <Button
        android:id="@+id/buttonLogIt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/log_it"/>

</LinearLayout>
```



Qual é a cola entre a interface XML e o código Java?



# Os Nomes dos Componentes

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    editText = (EditText) findViewById(R.id.editText1);
    button = (Button) findViewById(R.id.buttonLogIt);
}
```

Falta ainda programar  
o evento. Como fica  
isso?





# Mais um evento de botão

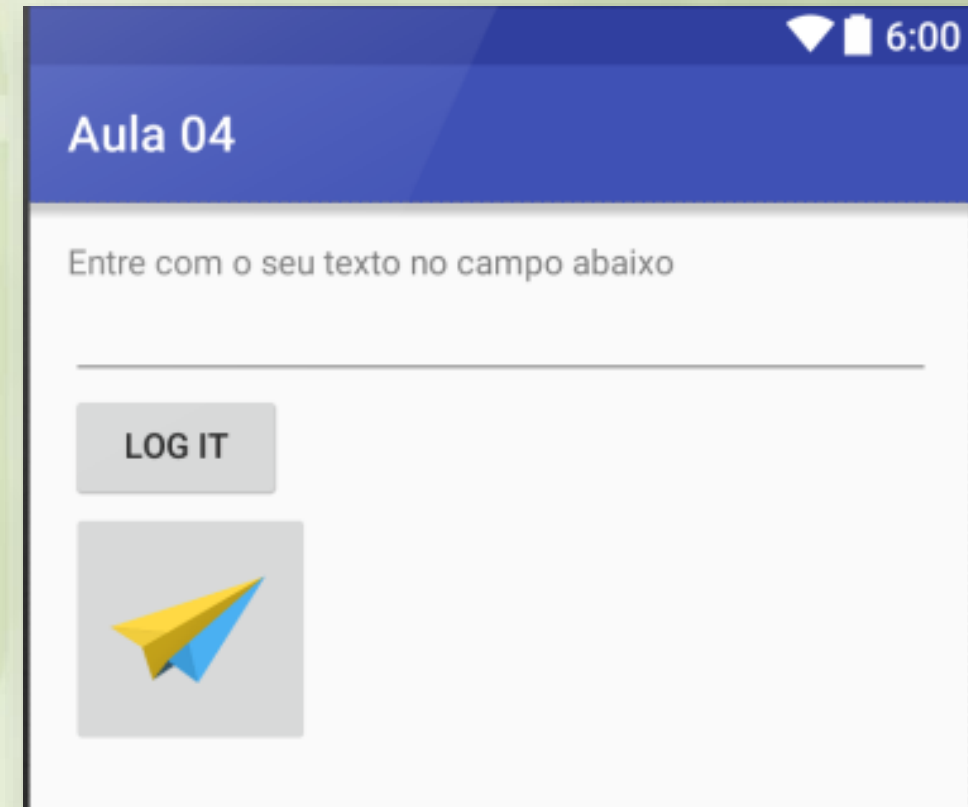
```
private final Button.OnClickListener  
    btnLogItOnClick = new Button.OnClickListener() {  
    public void onClick(View v) {  
        String texto = editText.getText().toString();  
        // imprimir o texto na saída do Log  
        Log.v("Aula04", texto);  
        editText.setText("");  
    }  
};
```

1. O que significa os parâmetros de Log.v?
2. O que esta linha está fazendo?



# Botões com imagens

```
<ImageButton  
    android:src="@drawable/ic_send"  
    android:scaleType="fitCenter"  
    android:layout_width="100dp"  
    android:layout_height="100dp" />
```



# Caixas de marcação

<CheckBox

```
    android:id="@+id/checkbox1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/checkbox_caixa_desmarcada"
    android:checked="false" />
```

Como implementar a camada de controle dessa aplicação?

☐ CheckBox: caixa desmarcada

☒ CheckBox: caixa marcada



# Eventos de marcação

```
checkBox.setOnClickListener(new CheckBox.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        if (checkBox.isChecked()) {  
            checkBox.setText(R.string.checkbox_caixa_marcada);  
        }  
        else {  
            checkBox.setText(R.string.checkbox_caixa_desmarcada);  
        }  
    }  
});
```



# Botões de rádio

```
<RadioGroup
    android:id="@+id/radioGroup1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">
    <RadioButton
        android:id="@+id/radioButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="radioOnClick"
        android:text="@string/radiobutton1" />
    <RadioButton
        android:id="@+id/radioButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="radioOnClick"
        android:text="@string/radiobutton2" />
    <RadioButton
        android:id="@+id/radioButton3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="radioOnClick"
        android:text="@string/radiobutton3" />
</RadioGroup>
```

Como implementar a camada de controle dessa aplicação?

☐ RadioButton 1

☐ RadioButton 2

☐ RadioButton 3

RadioGroup: Nada escolhido ainda

☒ RadioButton 1

☐ RadioButton 2

☐ RadioButton 3

RadioGroup: RadioButton 1

☐ RadioButton 1

☒ RadioButton 2

☐ RadioButton 3

RadioGroup: RadioButton 2

☐ RadioButton 1

☐ RadioButton 2

☒ RadioButton 3

RadioGroup: RadioButton 3

# Botões de rádio

```
private CheckBox checkBox;  
private TextView textViewElementoEscolhido;  
  
@Override  
protected void onCreate(Bundle savedInstanceState) {  
  
    ...  
  
    checkBox = (CheckBox) findViewById(R.id.checkBox1);  
    textViewElementoEscolhido = (TextView) findViewById(R.id.textViewElementoEscolhido);  
  
    ...  
}  
  
public void radioButtonOnClick(View v) {  
    RadioButton radioButton = (RadioButton) v;  
    textViewElementoEscolhido.setText(radioButton.getText());  
}
```

# Spinners

```
<Spinner  
    android:id="@+id/spinner1"  
    android:layout_width="150dp"  
    android:layout_height="wrap_content"/>
```

Athos ▼

Athos

Porthos

Aramis



# Spinners

```
List<String> lsSpinner = new ArrayList<>();
lsSpinner.add("Athos");
lsSpinner.add("Porthos");
lsSpinner.add("Aramis");
ArrayAdapter<String> arrayAdapter
    = new ArrayAdapter<String>(this,
        android.R.layout.simple_spinner_item,
        lsSpinner);
arrayAdapter.setDropDownViewResource(
    android.R.layout.simple_dropdown_item_1line);
spinner.setAdapter(arrayAdapter);
spinner.setOnItemSelectedListener(new Spinner.OnItemSelectedListener() {
    @Override
    public void onItemSelected(
        AdapterView<?> parent, View view, int position, long id) {
        // ???
    }
    @Override
    public void onNothingSelected(AdapterView<?> parent) {
        // ???
    }
});
```

O que faz esse bloco de código?

E esse bloco de código, o que faz?

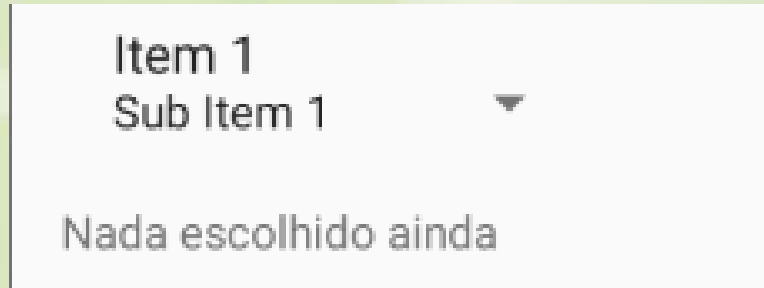




# Eventos de Spinner

Adicione um controlador de eventos a esta Spinner:

- Modifique a visão para incluir uma caixa de texto
- Cada item selecionado deve ser mostrado na caixa de texto



# Múltiplas Atividades

- É possível desenvolver múltiplas atividades na mesma aplicação
- Novas atividades são declaradas em AndroidManifest.xml
- Usando o Android Studio, é possível escolher qual a atividade a ser lançada no emulador
  - Run, Edit configurations..., Launch.



# AndroidManifest.xml

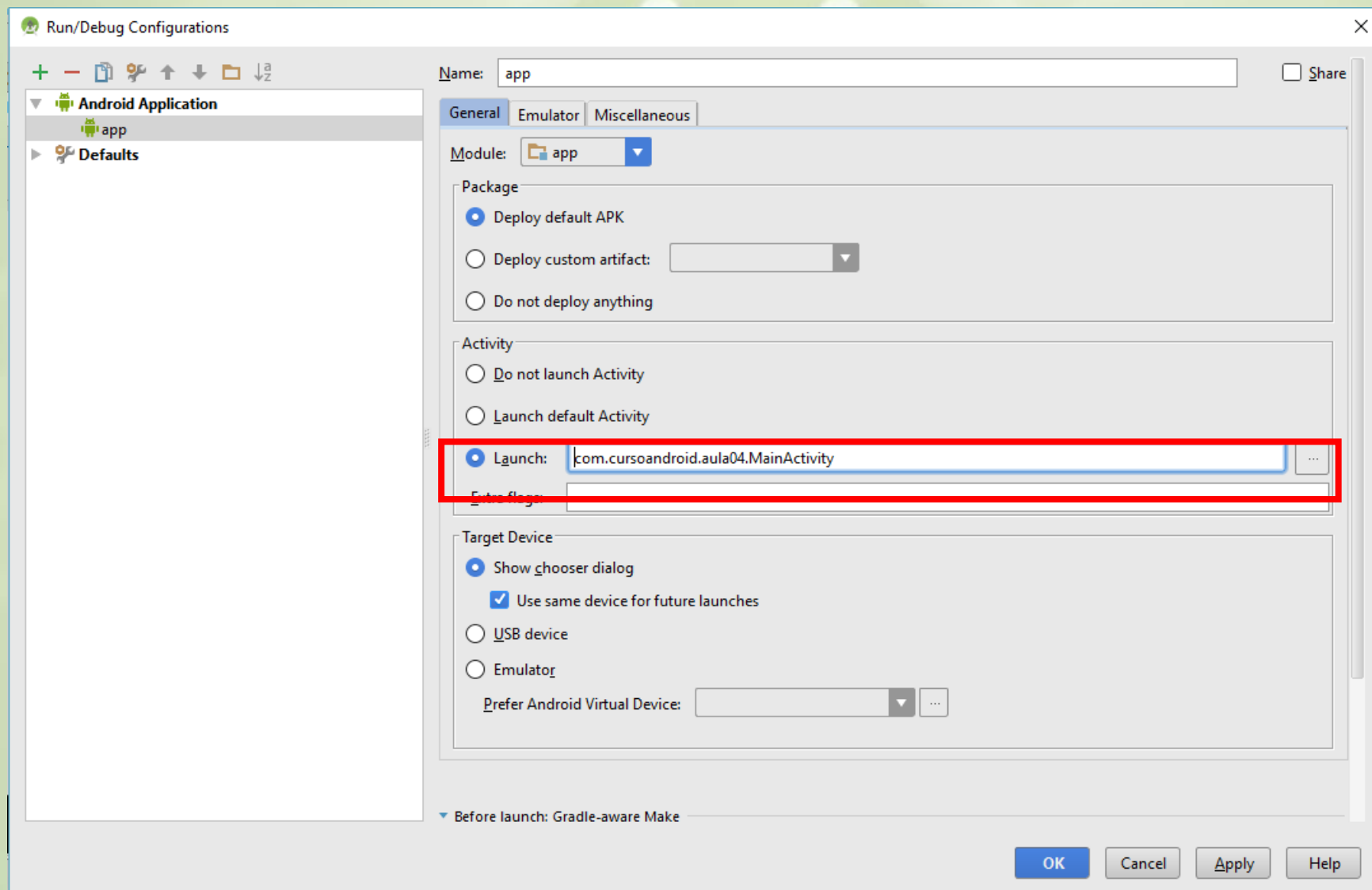
```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/AppTheme" >
    <activity android:name=".MainActivity" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity android:name=".Gallery" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
```



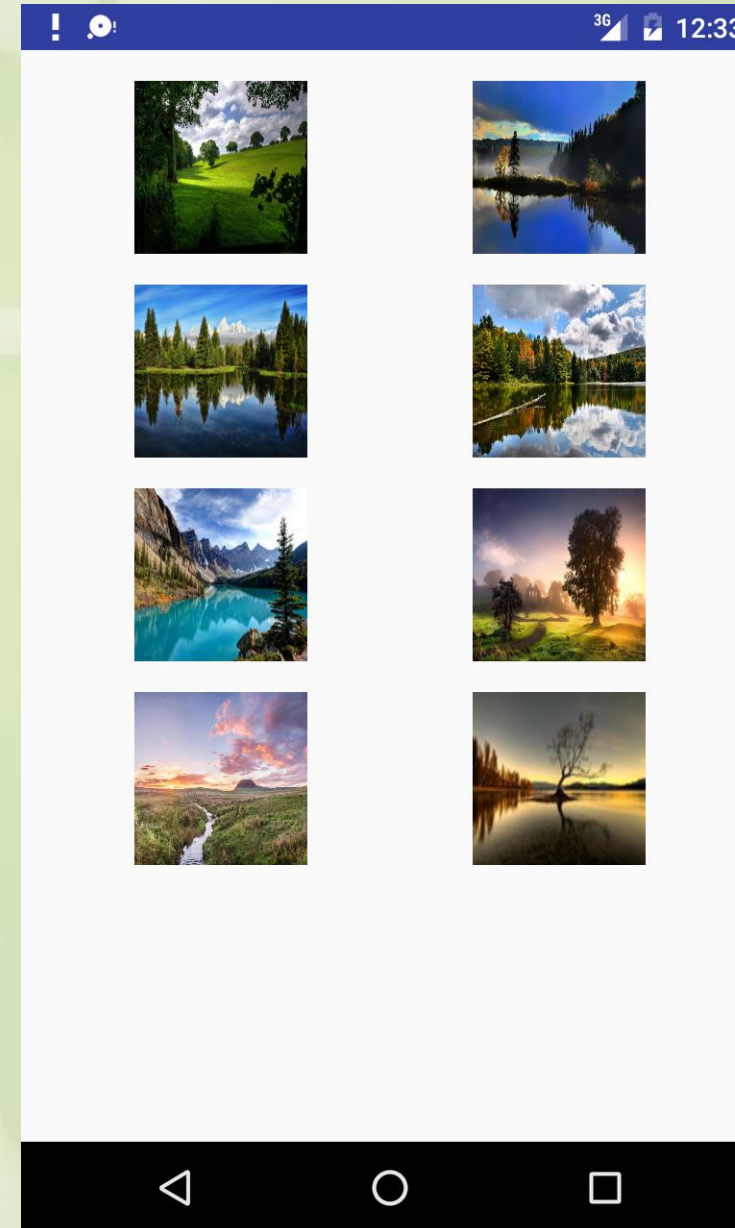
# A atividade a ser lançada no emulador



# Galerias de imagens

## Lista de fotos

- Crie uma galeria de imagens, tal que seja possível escolher alguma dentre as fotos mostradas
- Cada vez que uma foto é escolhida, seu índice na galeria deve ser mostrado no Toast do aparelho



# Galerias

- O Android possui um elemento gráfico que pode ser criado para criar galerias de imagens

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <GridView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"/>

</LinearLayout>
```



# Inicialização da galeria

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_gallery);
    GridView gridView = (GridView) findViewById(R.id.gridView);
    gridView.setAdapter(new ImageAdapter(this));
    gridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            Toast.makeText(Gallery.this, "" + position, Toast.LENGTH_LONG).show();
        }
    });
}
```

O que é esta classe  
Toast?



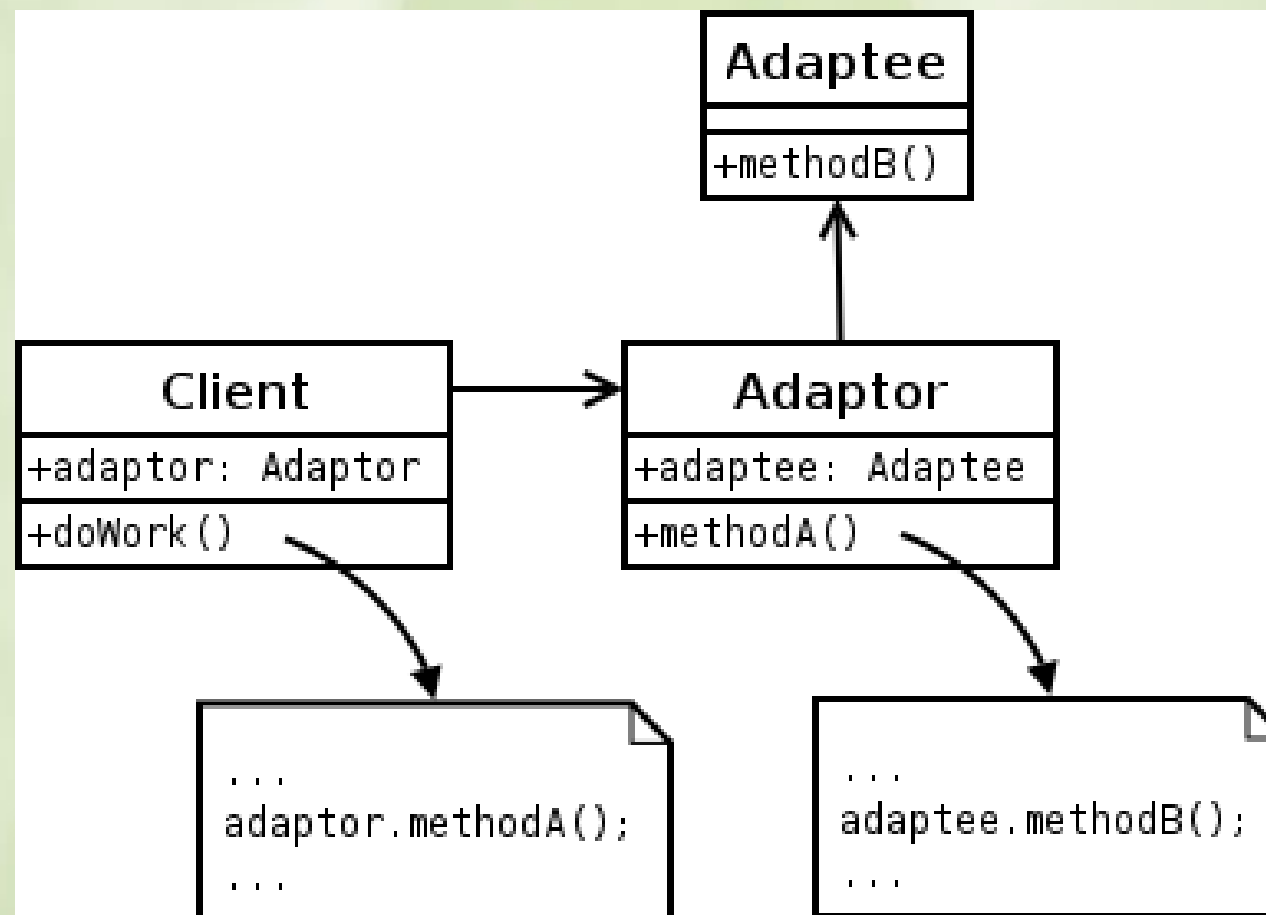
# Adaptadores

- Adapter é um padrão de projetos
  - Aliás, um dos mais simples que existem
- Um adaptador faz a adequação entre duas interfaces
  - Quais interfaces precisam ser adaptadas, em nosso caso?





# Adaptadores em UML



# ImageAdapter

```
public class ImageAdapter extends BaseAdapter {  
    @Override  
    public int getCount() { ... }  
    @Override  
    public Object getItem(int position) { ... }  
    @Override  
    public long getItemId(int position) { ... }  
    @Override  
    public View getView(int position, View convertView, ViewGroup parent) { ... }  
}
```

Como implementar  
este adaptador?



# ImageAdapter

```
public class ImageAdapter extends BaseAdapter {  
    @Override  
    public int getCount() { ... }  
    @Override  
    public Object getItem(int position) { ... }  
    @Override  
    public long getItemId(int position) { ... }  
    @Override  
    public View getView(int position, View convertView, ViewGroup parent) { ... }  
}
```

- Número de itens representados pelo adaptador
- Objeto em uma certa posição
- Visão que mostrará o objeto armazenado na dada posição



# A lista de imagens

```
private Integer[] mImageIds = {  
    R.drawable.gallery_photo_1,  
    R.drawable.gallery_photo_2, R.drawable.gallery_photo_3,  
    R.drawable.gallery_photo_4, R.drawable.gallery_photo_5,  
    R.drawable.gallery_photo_6, R.drawable.gallery_photo_7,  
    R.drawable.gallery_photo_8 };  
  
public ImageAdapter(Context context) {  
    mContext = context;  
}  
@Override  
public int getCount() { return mImageIds.length; }  
@Override  
public Object getItem(int position) { return mImageIds[position]; }  
@Override  
public long getItemId(int position) { return position; }
```



# E o mais importante...

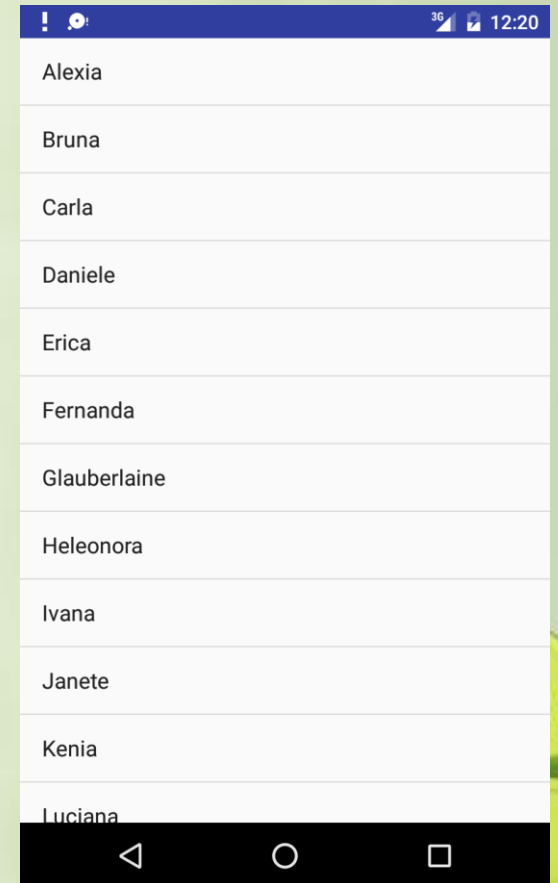
```
@Override
public View getView(int position, View convertView, ViewGroup parent) {
    ImageView i = new ImageView(mContext);
    i.setImageResource(mImageIds[position]);
    i.setScaleType(ImageView.ScaleType.FIT_XY);
    i.setLayoutParams(new LinearLayout.LayoutParams(180, 180));
    i.setBackgroundResource(mGalleryItemBackground);
    return i;
}
```



# Listas

- Uma das visões mais utilizadas são as listas
  - Existe uma atividade que já incorpora o layout de listas: trata-se de ListActivity

```
public class List1 extends ListActivity {  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setListAdapter(new ArrayAdapter<String>(this,  
            android.R.layout.simple_list_item_1, mStrings));  
        getListView().setTextFilterEnabled(true);  
    }  
    private String[] mStrings = {"Alexia", "Bruna", "Carla",  
        "Daniele", "Erica", "Fernanda", "Glauberlaine",  
        "Heleonora", "Ivana", "Janete", "Kenia", "Luciana",  
        "Marta", "Norma", "Osma", "Patricia"};  
}
```



# Eventos de listas

- Uma das visões mais utilizadas são as listas
  - Existe uma atividade que já incorpora o layout de listas: trata-se de `ListActivity`

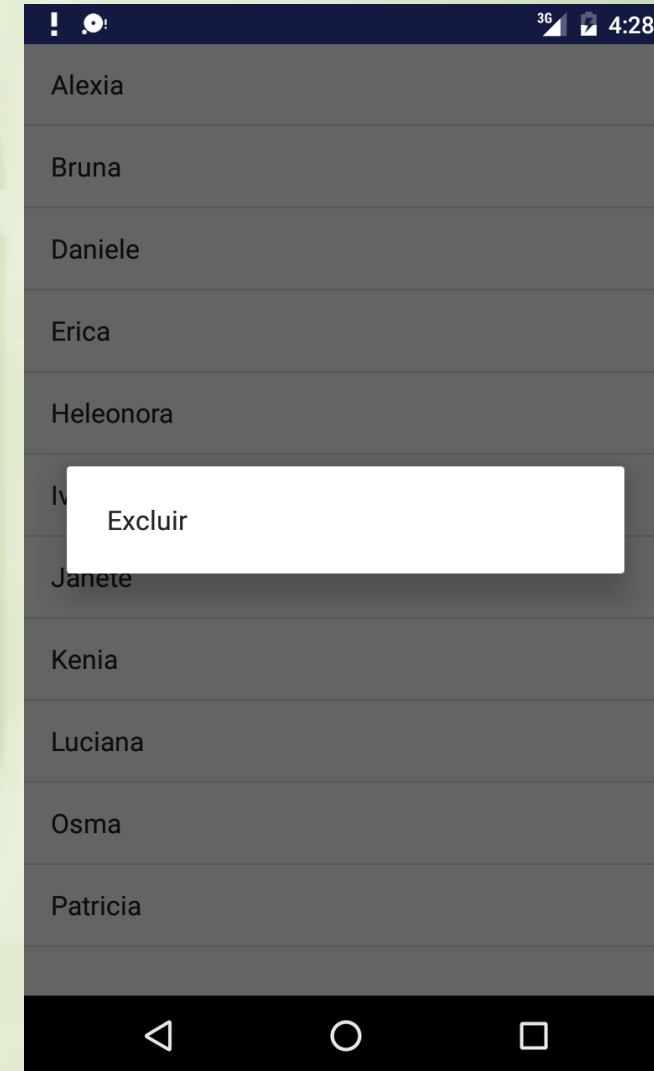
Adicione a funcionalidade de remover elementos de sua aplicação.

Discuta a estratégia escolhida.

Será que essa estratégia pode ser estendida no futuro?

# Menu de contexto

- Menus de contexto aparecem em Janelas pop-up
- Podemos ativar o menu de contexto de uma lista se clicarmos sobre um de seus itens e pressionarmos o ponteiro durante um tempo
- Mas, como adicionar um menu de contexto à lista?





# Menu de contexto

E como tratar o evento produzido pelo menu?

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setListAdapter(new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, mStrings));
    registerForContextMenu(getListView());
    getListView().setTextFilterEnabled(true);
    initList();
}

private final int del = 1;
private List<String> mStrings = new ArrayList<>();
public final void onCreateContextMenu(final ContextMenu menu,
                                     final View v,
                                     final ContextMenu.ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    menu.add(Menu.NONE, del, Menu.NONE, R.string.excluir);
}
```

**Iniciar lista para testes**

# Tratando eventos de contexto

```
public final boolean onContextItemSelected(final MenuItem item) {
    AdapterView.AdapterContextMenuInfo info =
        (AdapterView.AdapterContextMenuInfo) item.getMenuInfo();
    ListAdapter adapter = getListAdapter();
    String e = (String) adapter.getItem((int) info.id);
    switch (item.getItemId()) {
        case del:
            mStrings.remove(e);
            ((BaseAdapter) getListView().getAdapter()).notifyDataSetChanged();
            return true;
        default:
            return super.onContextItemSelected(item);
    }
}
```

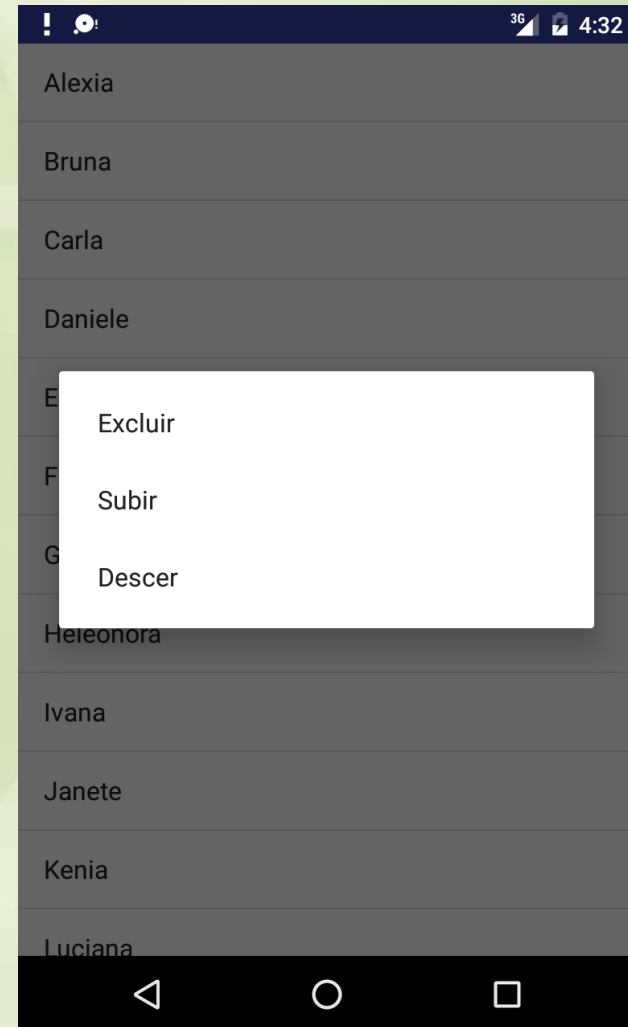
Como é feita a amarração  
entre o evento escolhido  
e o evento tratado?



# Estendendo a aplicação

## Múltiplos eventos

- Adicione um evento “Subir” para mover o item selecionado para a primeira posição da lista
- Adicione um evento “Descer” para mover o item selecionado para a última posição da lista



# Múltiplos eventos

```
private final int del = 1;
private final int up = 2;
private final int down = 3;

public final void onCreateContextMenu(final ContextMenu menu,
                                      final View v,
                                      final ContextMenu.ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    menu.add(Menu.NONE, del, Menu.NONE, R.string.excluir);
    menu.add(Menu.NONE, del, Menu.NONE, R.string.subir);
    menu.add(Menu.NONE, del, Menu.NONE, R.string.descer);
}
```



# Múltiplos eventos

```
public final boolean onContextItemSelected(final MenuItem item) {
    AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo) item.getMenuInfo();
    ListAdapter adapter = getListAdapter();
    String e = (String) adapter.getItem((int) info.id);
    switch (item.getItemId()) {
        case del:
            mStrings.remove(e);
            break;
        case up:
            mStrings.remove(e);
            mStrings.add(0, e);
            break;
        case down:
            mStrings.remove(e);
            mStrings.add(e);
            break;
        default:
            return super.onContextItemSelected(item);
    }
    this.setListAdapter(new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, mStrings));
    return true;
}
```

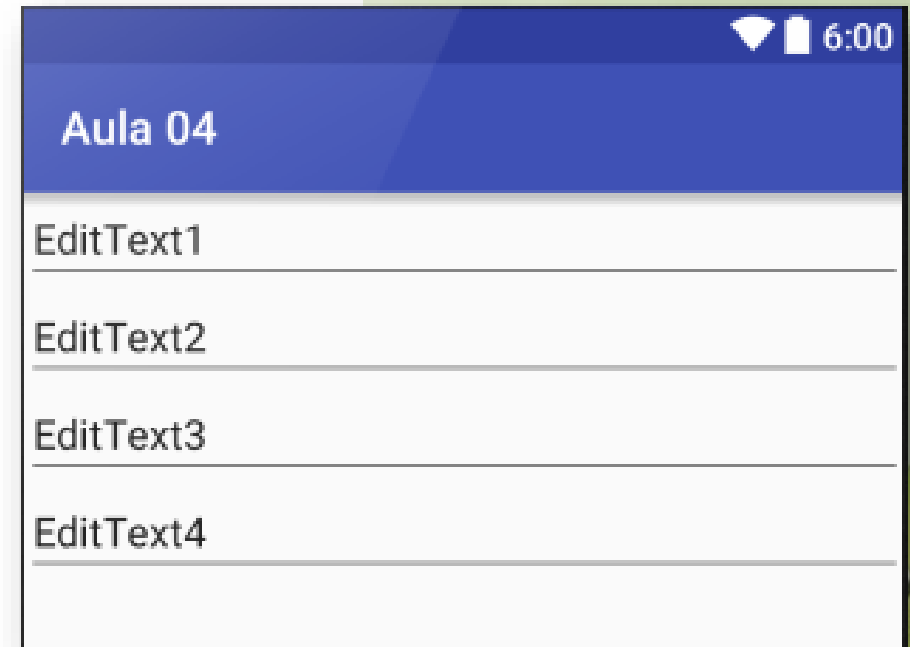
# Mais um pouco sobre layouts

- O Android usa layouts para controlar a aparência das aplicações
- Existem vários tipos de layouts
  - `FrameLayout`
  - `LinearLayout`
  - `TableLayout`
  - `AbsoluteLayout`
  - `RelativeLayout`
  - Etc
- Para que serve cada um desses formatos?



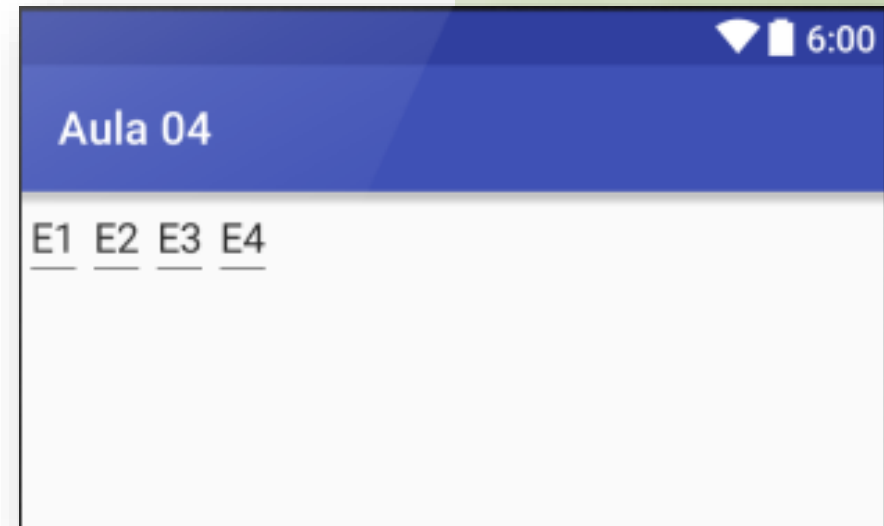
# LinearLayout

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="EditText1" />
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="EditText2" />
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="EditText3" />
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="EditText4"
    />
</LinearLayout>
```



# LinearLayout

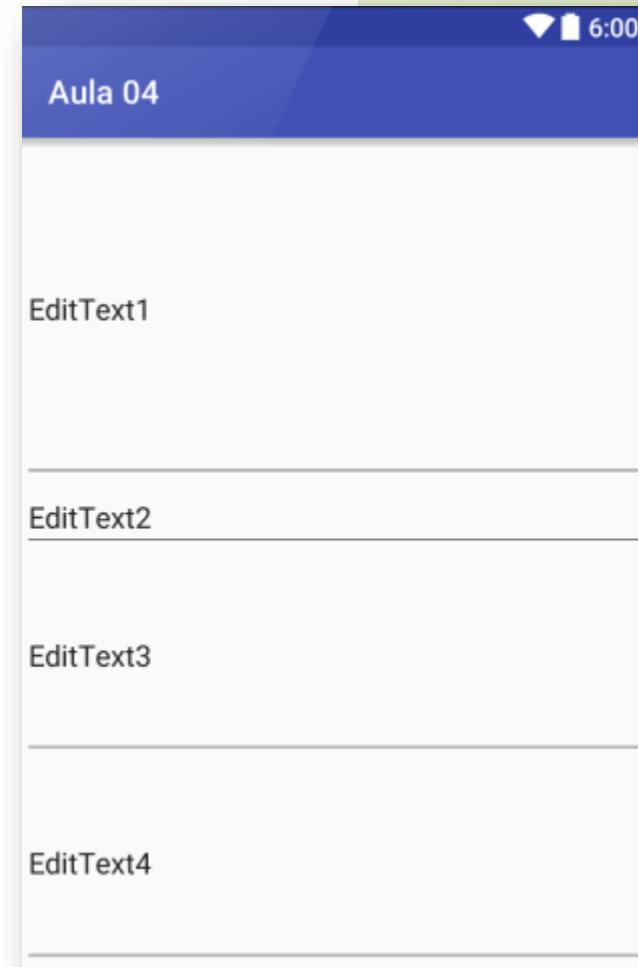
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="E1" />
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="E2" />
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="E3" />
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="E4" />
</LinearLayout>
```





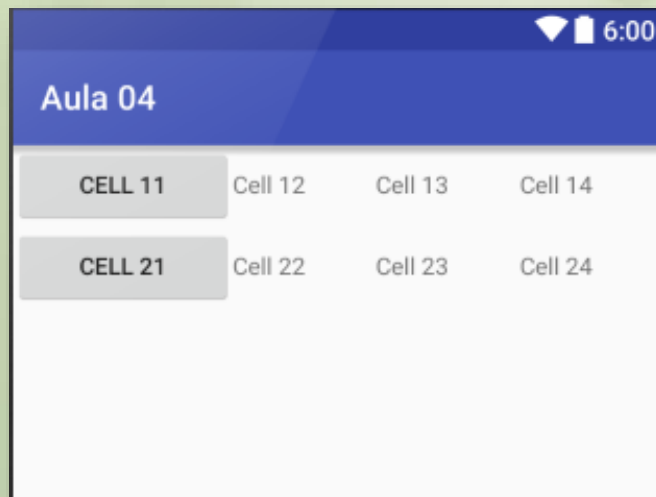
# Pesos

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="EditText1"
        android:layout_weight="1"/>
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="EditText2" />
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="EditText3"
        android:layout_weight="0.5"/>
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="EditText4"
        android:layout_weight="0.5"/>
</LinearLayout>
```



# TableLayout

- Tabelas permitem que componentes gráficos sejam armazenados em linhas e colunas
- É possível determinar o peso de cada item?



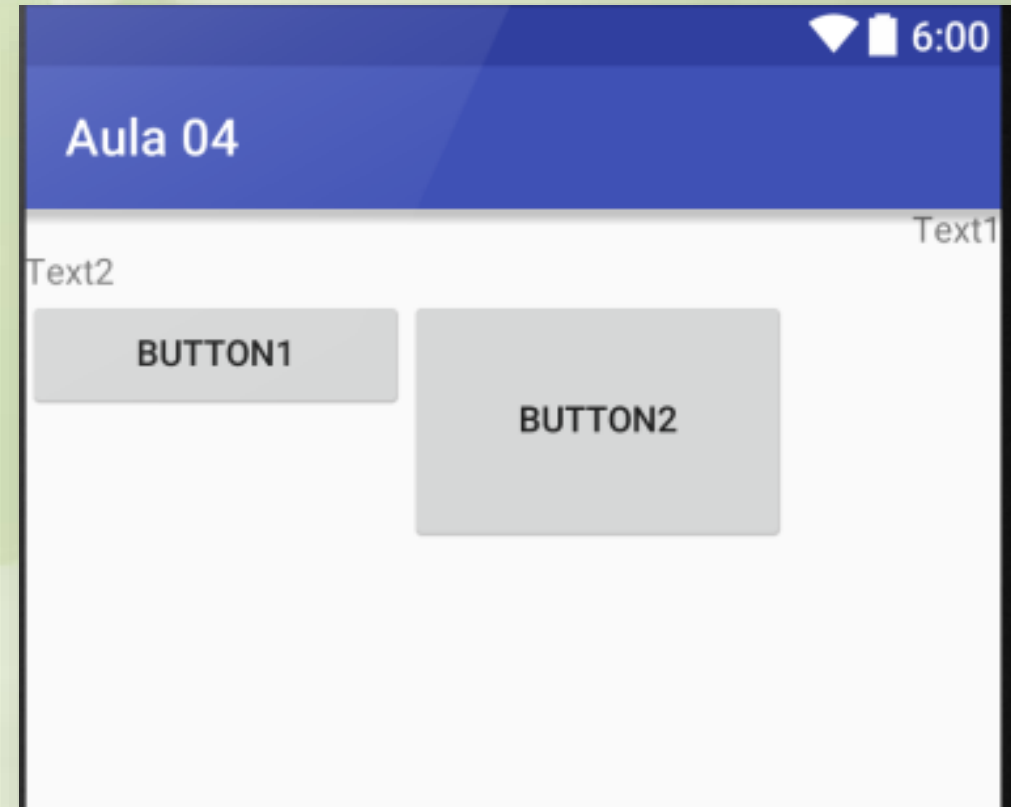
```
<?xml version="1.0" encoding="utf-8" ?>
<TableLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/tblJobs"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content">
    <TableRow
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <Button android:text="Cell 11"
            android:id="@+id/btnCell11"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>
        <TextView
            android:id="@+id/txtCell12"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Cell 12"
            android:layout_weight="1"/>
        </TableRow>
    </TableLayout>
```



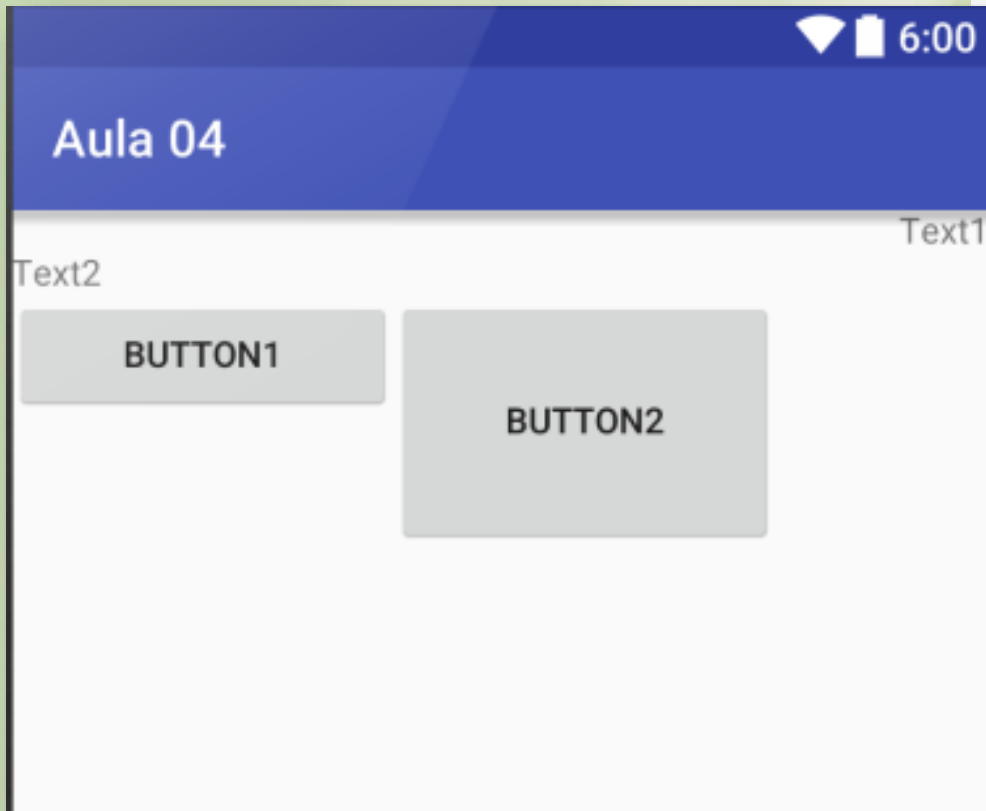
# RelativeLayout

- É possível especificar a posição de um elemento gráfico com relação a outro, usando layouts relativos

```
<Button
    android:id="@+id/btnButton1"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:text="Button1"
    android:layout_below="@+id/txtText2"/>
<Button
    android:id="@+id/btnButton2"
    android:layout_width="150dp"
    android:layout_height="100dp"
    android:text="Button2"
    android:layout_toRightOf="@+id/btnButton1"
    android:layout_alignTop="@+id/btnButton1"/>
```



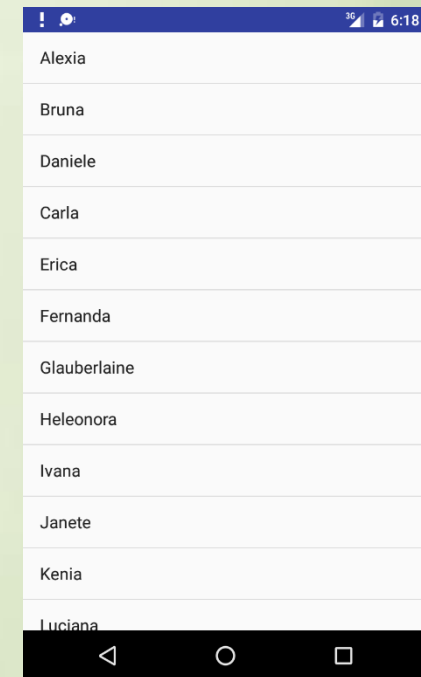
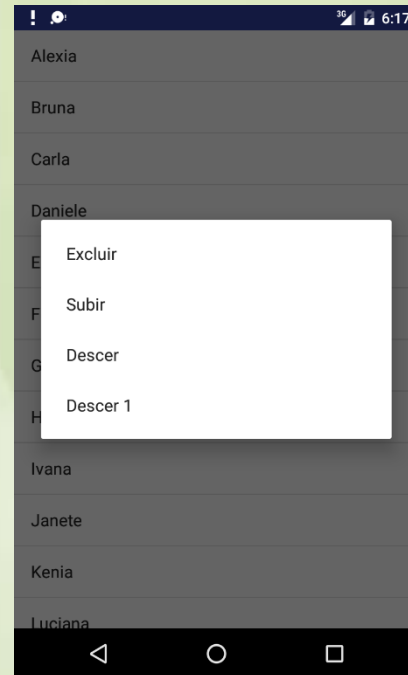
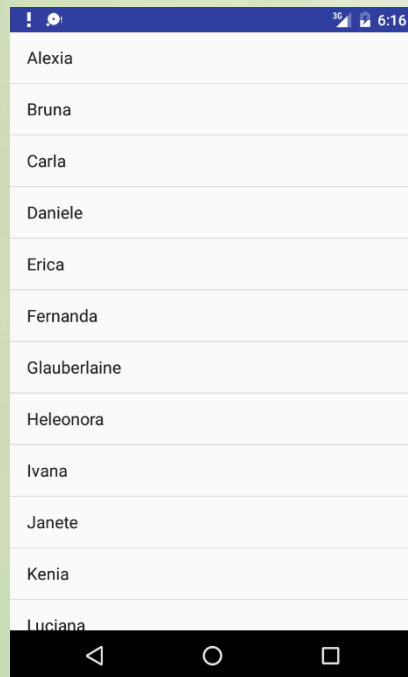
# RelativeLayout



```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView
        android:id="@+id/txtText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Text1"
        android:gravity="top"
        android:layout_alignParentRight="true"/>
    <TextView
        android:id="@+id/txtText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Text2"
        android:layout_below="@+id/txtText1"/>
    <Button
        android:id="@+id/btnButton1"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:text="Button1"
        android:layout_below="@+id/txtText2"/>
    <Button
        android:id="@+id/btnButton2"
        android:layout_width="150dp"
        android:layout_height="100dp"
        android:text="Button2"
        android:layout_toRightOf="@+id/btnButton1"
        android:layout_alignTop="@+id/btnButton1"/>
</RelativeLayout>
```

# Incrementando a nossa lista

- Modifique a atividade List1 para que seja possível mover um item uma posição para baixo. O último item não pode ser movido, mas a sua aplicação não deve para de funcionar se o usuário tentar movê-lo



# Dúvidas

