SESIÓN 4

Listas (ListView)

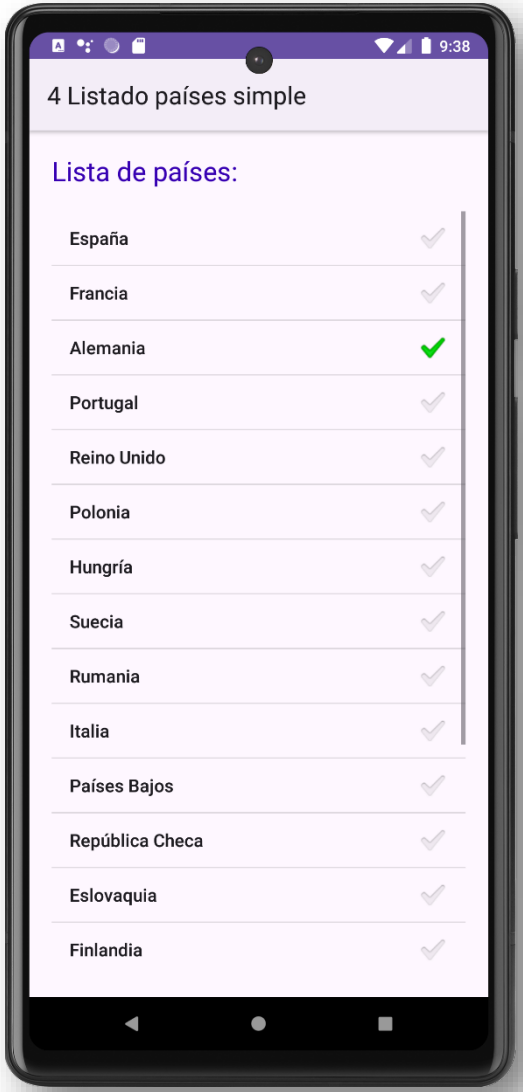
Personalización de adaptadores

Widget ListView

- **`ListView`** es un componente gráfico que muestra una lista vertical con *scroll* de elementos (`View`).
- Al igual que `Spinner`:
  - Usa un adaptador (`Adapter`) para obtener los datos que se mostrarán en la lista.
  - Hay que especificar el diseño con el que se mostrará cada ítem de la lista, pudiéndose usar uno predefinido en Android o bien diseñar uno propio.
- En su forma más básica presenta una lista de `String` (con *scroll*) y usa `ArrayAdapter` como adaptador.
- La lista puede ser de selección simple o múltiple. Esto se puede especificar por código o en el XML del `ListView`:

```
listView.setChoiceMode( ListView.CHOICE_MODE_MULTIPLE );
```

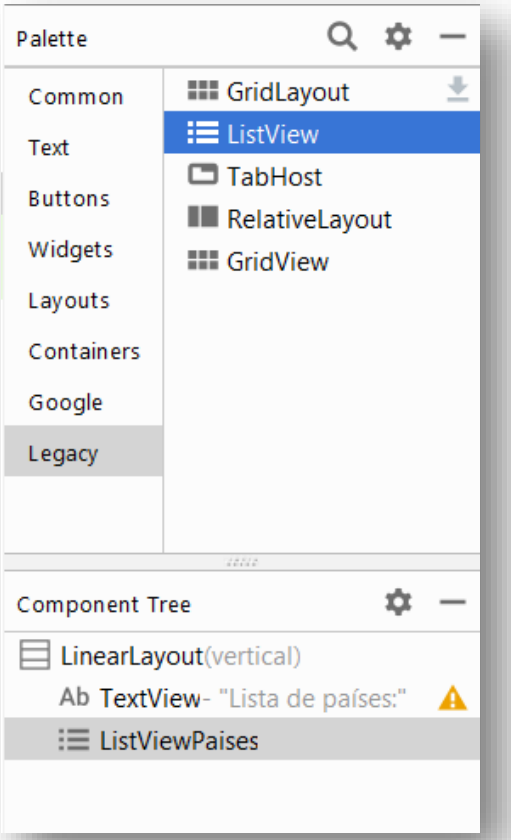
```
<ListView android:id="@android:id/list"
          android:choiceMode="singleChoice"
```

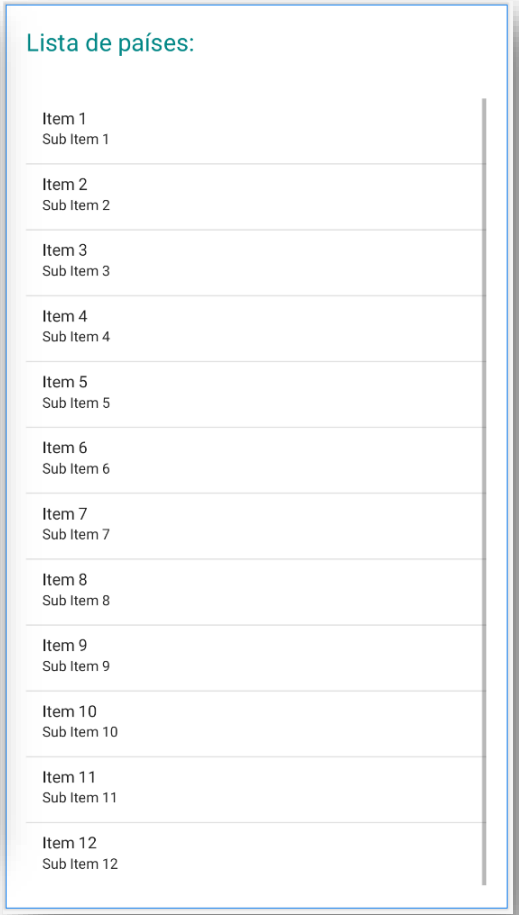


https://developer.android.com/reference/android/widget/ListView

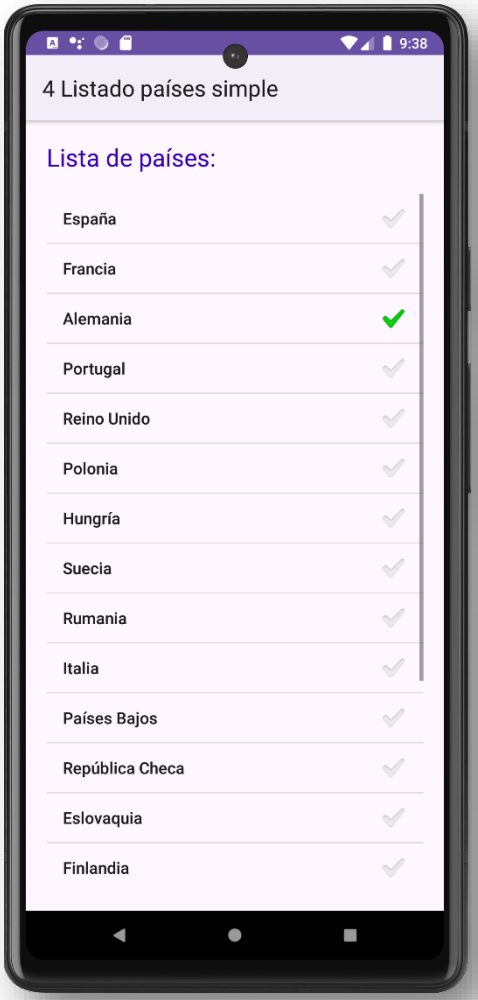
Ejemplo del diseño de una actividad que usa un ListView

```xml
<?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:layout_margin="20dp"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Lista de países:"
        android:textColor="@color/teal_700"
        android:textSize="24sp" />

    <ListView
        android:id="@+id/ListViewPaises"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

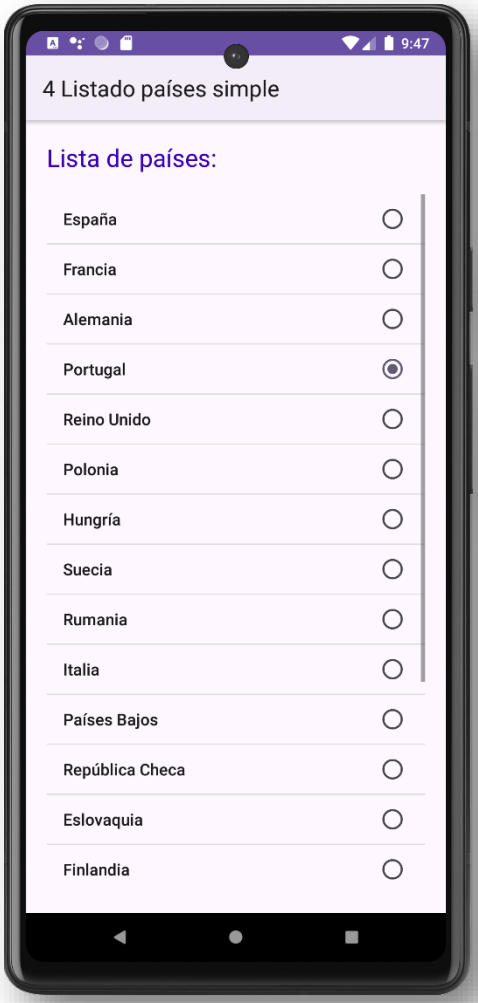




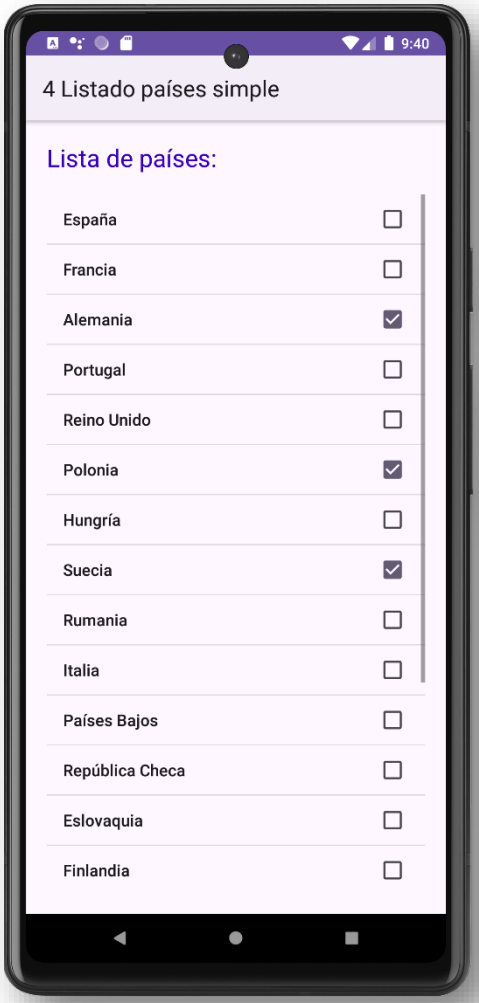
Algunos diseños de ítems de ListView predefinidos



`simple_list_item_checked`



`simple_list_item_single_choice`



`simple_list_item_multiple_choice`

Diseños predefinidos

- https://android.googlesource.com/platform/frameworks/base/+/master/core/res/res/layout/
- simple_list_item_checked.xml:

```
<CheckedTextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@android:id/text1"
    android:layout_width="match_parent"
    android:layout_height="?android:attr/listPreferredItemHeightSmall"
    android:textAppearance="?android:attr/textAppearanceListItemSmall"
    android:gravity="center_vertical"
    android:checkMark="?android:attr/textCheckMark"
    android:paddingStart="?android:attr/listPreferredItemPaddingStart"
    android:paddingEnd="?android:attr/listPreferredItemPaddingEnd" />
```

Ejemplo de código

Datos a visualizar en la lista

Adaptador: layout y array de datos

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

    final String[] paises = {"España", "Alemania", "Reino Unido", "Francia", "Italia"};


    // Crear ArrayAdapter de lista de países. Parámetros: layout del ListView y datos.
    // Si los datos están en la variable paises, crear ArrayAdapter usando una de siguientes:
    ArrayAdapter paisesAdaptador = new ArrayAdapter(this,
                                        android.R.layout.simple_list_item_checked, paises);
    ArrayAdapter paisesAdaptador = new ArrayAdapter(this,
                                       android.R.layout.simple_list_item_single_choice, paises);
    ArrayAdapter paisesAdaptador = new ArrayAdapter(this,
                                       android.R.layout.simple_list_item_multiple_choice, paises);
    // Si los datos están en el xml paises_array.xml:
    ArrayAdapter paisesAdaptador = ArrayAdapter.createFromResource(this,
                           R.array.paises_array,android.R.layout.simple_list_item_multiple_choice)

    // Obtener referencia al ListView definido en el layout de la actividad activity_main.xml:
    final ListView listadoPaises = findViewById(R.id.ListViewPaises);
    // Asignar el ArrayAdapter al ListView:
    listadoPaises.setAdapter(paisesAdaptador);

    // Definir el modo del ListView (elección única o múltiple). Uno de los tres siguientes:
    listadoPaises.setChoiceMode(ListView.CHOICE_MODE_SINGLE);     // Única
    listadoPaises.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);   // Múltiple
}
```

Recoger elemento seleccionado (1)

Dos formas:

1. Mediante el método **`getSelectedItem()`** de **`ListView`**
2. Estableciendo un manejador en el **`ListView`** con el método **`setOnItemClickListener()`**, donde se define la acción que se llevará a cabo al hacer clic en un ítem de un `ListView`. Hay que implementar el método **`onItemClick()`**

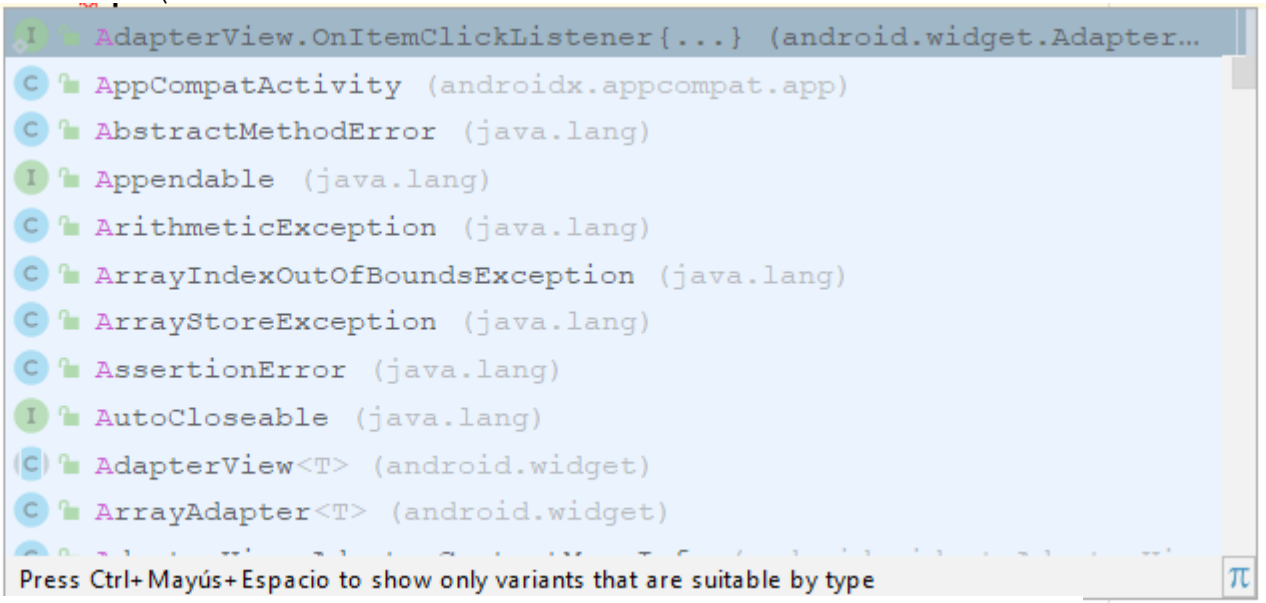
```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

     // parent: adaptador usado en el ListView donde se ha hecho clic
     // view: la vista del ListView el que se ha hecho clic
     // position: la posición de la vista (en el adaptador)
     // id: la posición original en la fuente de datos
         }
     });
```

https://developer.android.com/reference/android/widget/AdapterView.OnItemClickListener?hl=es

Recoger elemento seleccionado (2)

Android Studio ofrece autocompletado:

```
listView.setOnItemClickListener(new A
```




```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

    }
});
```

Recoger elemento seleccionado (3)

Ejemplo de recoger el texto de un ítem en selección simple:

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView adapterView, View view,
                                                    int position, long id) {
        String item;
        item = listView.getItemAtPosition(position).toString();

    }
});
```

SparseBooleanArray (1)

- Similar a un `HashMap` que mapea enteros a *boolean* permitiendo huecos en los índices
- Se puede iterar con los métodos:
  - `int keyAt (int index)`: devuelve la clave de la posición indicada
  - `boolean valueAt (int index)`: devuelve el valor de la posición indicada

Posición	0	1	2	3
Clave	1	3	5	7
Valor	true	false	false	true

https://developer.android.com/reference/android/util/SparseBooleanArray

SparseBooleanArray (2)

Posición	0	1
Clave	1	3
Valor	true	true

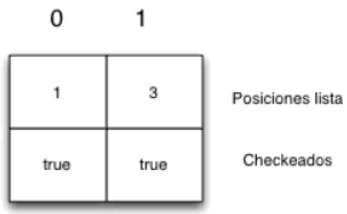
0	1	2
0	1	3
true	true	true

0	1	2
0	1	3
true	false	true

Recoger elemento seleccionado (4)

Ejemplo de recoger los textos de los ítems en selección múltiple:

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
  @Override
  public void onItemClick(AdapterView adapterView, View view,
                                                    int posicion, long id) {
     final StringBuilder items = new StringBuilder();
     SparseBooleanArray seleccionados=listView.getCheckedItemPositions();
     for (int i = 0; i < seleccionados.size(); i++) {
          if ( seleccionados.valueAt(i) ) {
              int pos = seleccionados.keyAt(i);
              items.append("\n");
              items.append(listView.getItemAtPosition(pos).toString());
          }
     }
  }
});
```



https://developer.android.com/reference/android/widget/AbsListView#getCheckedItemPositions()

Cambiar el color de fondo del elemento seleccionado

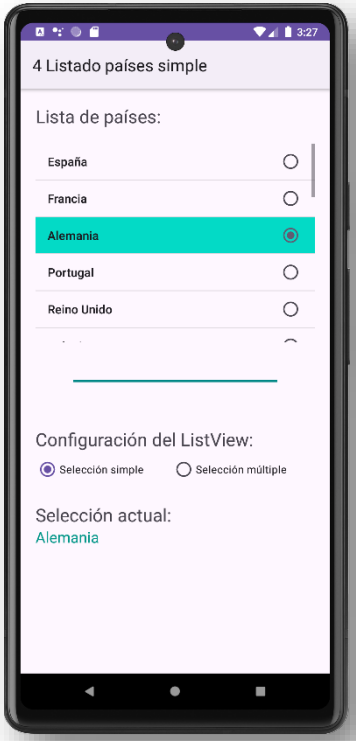
- En el XML del ListView:

```
<ListView
    android:id="@+id/ListViewPaises"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:listSelector="@color/design_default_color_secondary" />
```

- Por código es posible hacerlo, pero habría que sobrescribir el método `getView()` que se cuentan más adelante cuando se vea el adaptador personalizado

Ejercicio 1

- Crea un proyecto nuevo, con las siguientes características:
  - Nombre: “4 Listado países simple”
  - Paquete: dam.listadoPaisesSimple
  - Directorio: 4-ListadoPaisesSimple
- La actividad principal debe implementar un `ListView` de un listado de países con la opción de elegir si la selección es simple o múltiple
- En la parte inferior se debe mostrar el país (o países) seleccionado/s del listado
- Debe haber el suficiente número de países para que salga el *scroll*

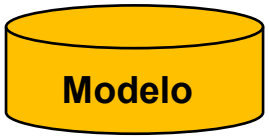


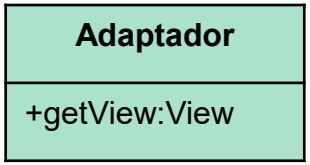
Pistas:

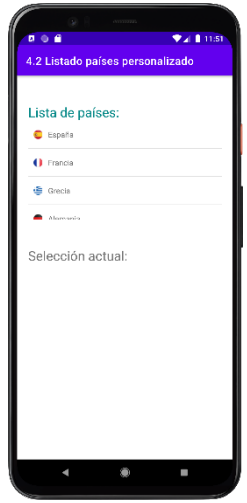
- Para obtener el listado de países, pídeselo a la IA y que te los devuelva en formato que necesitas para java
- Para cambiar de un listado de simple a uno múltiple hay que cambiar de adaptador, pues en la definición de adaptador se dice el diseño de los elementos

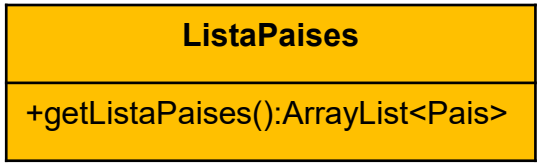
Adaptador personalizado (1)

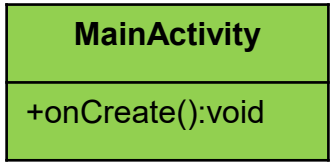
Patrón: Modelo-Vista-Controlador (MVC)

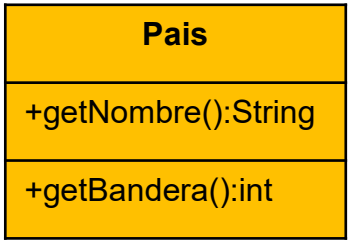












activity_main.xml

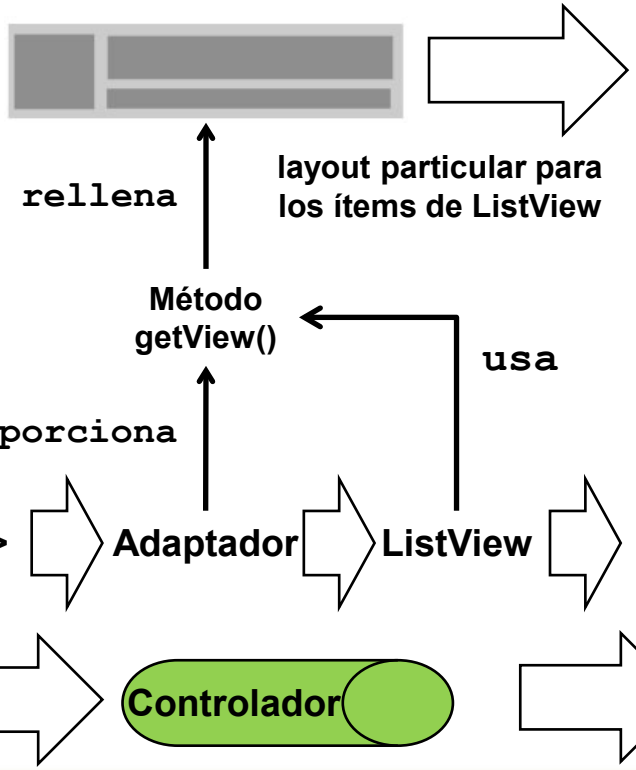
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.and
    xmlns:tools="http://schemas.android.com/too
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="20dp"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="30dp"
        android:text="Lista de paises:"
        android:textColor="@color/teal_700"
        android:textSize="24sp" />

    <ListView
        android:id="@+id/ListViewPaises"
        android:layout_width="match_parent"
        android:layout_height="180dp"/>
```

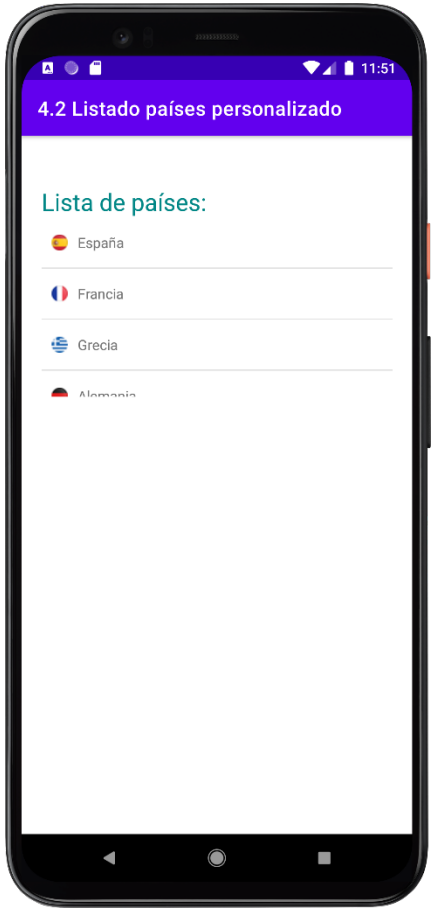
lista_paises_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:gravity="center_vertical">

    <ImageView
        android:id="@+id/paisBandera"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        tools:srcCompat="@tools:sample/avatars" />

    <TextView
        android:id="@+id/paisNombre"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="right"
        android:text="Nombre pais"/>
</LinearLayout>
```

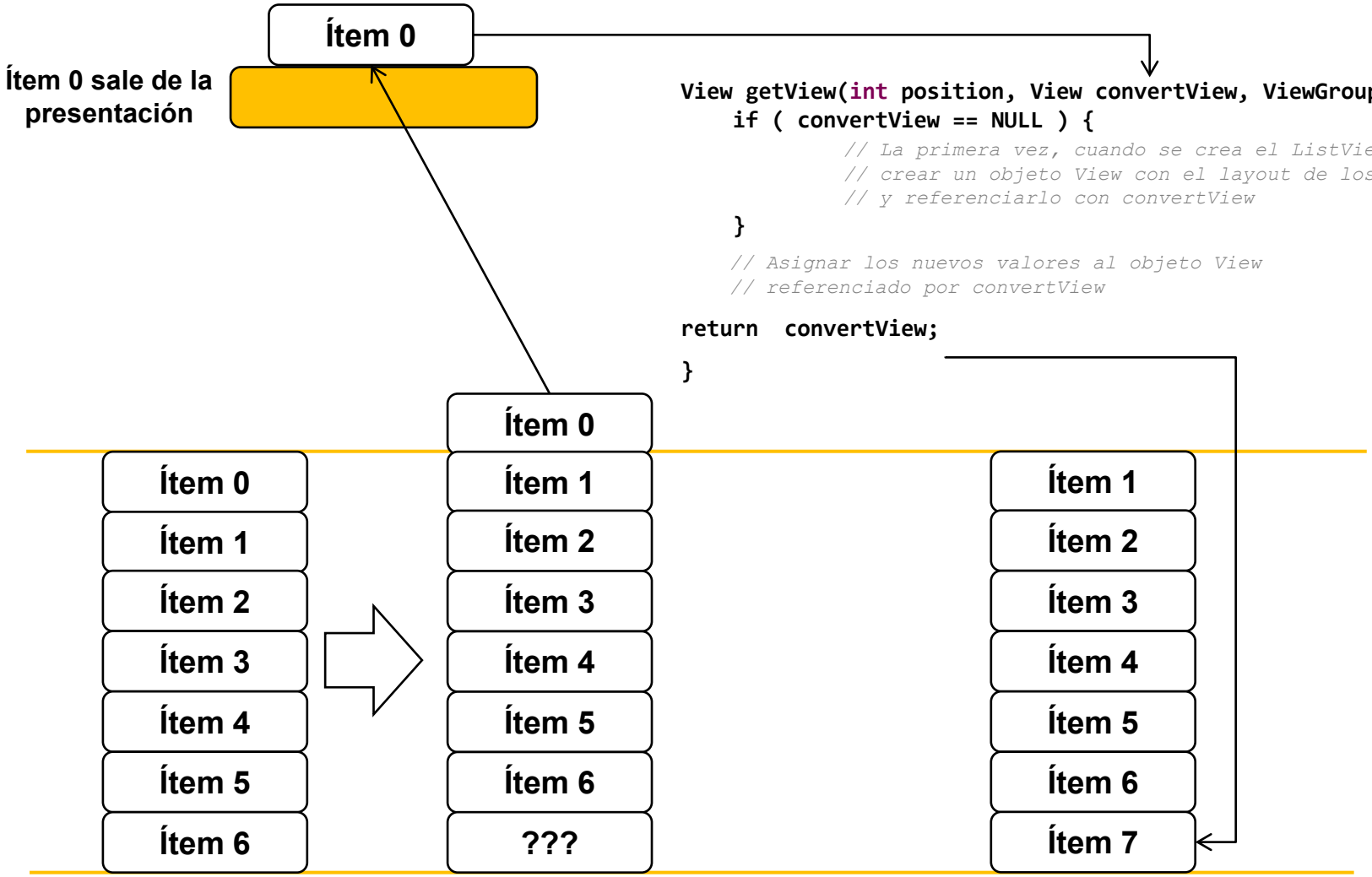
Adaptador personalizado (2)

- Un *adaptador personalizado* es una clase que hereda de un *adaptador básico* y proporciona, al menos, una implementación particular del método **`getView()`**.
- `ListView` (y otros componentes gráficos) usa `getView()` para formatear la vista de cada ítem.
- La implementación por defecto de `getView` en `ArrayAdapter<?>` usa `toString()` sobre cada elemento del array.

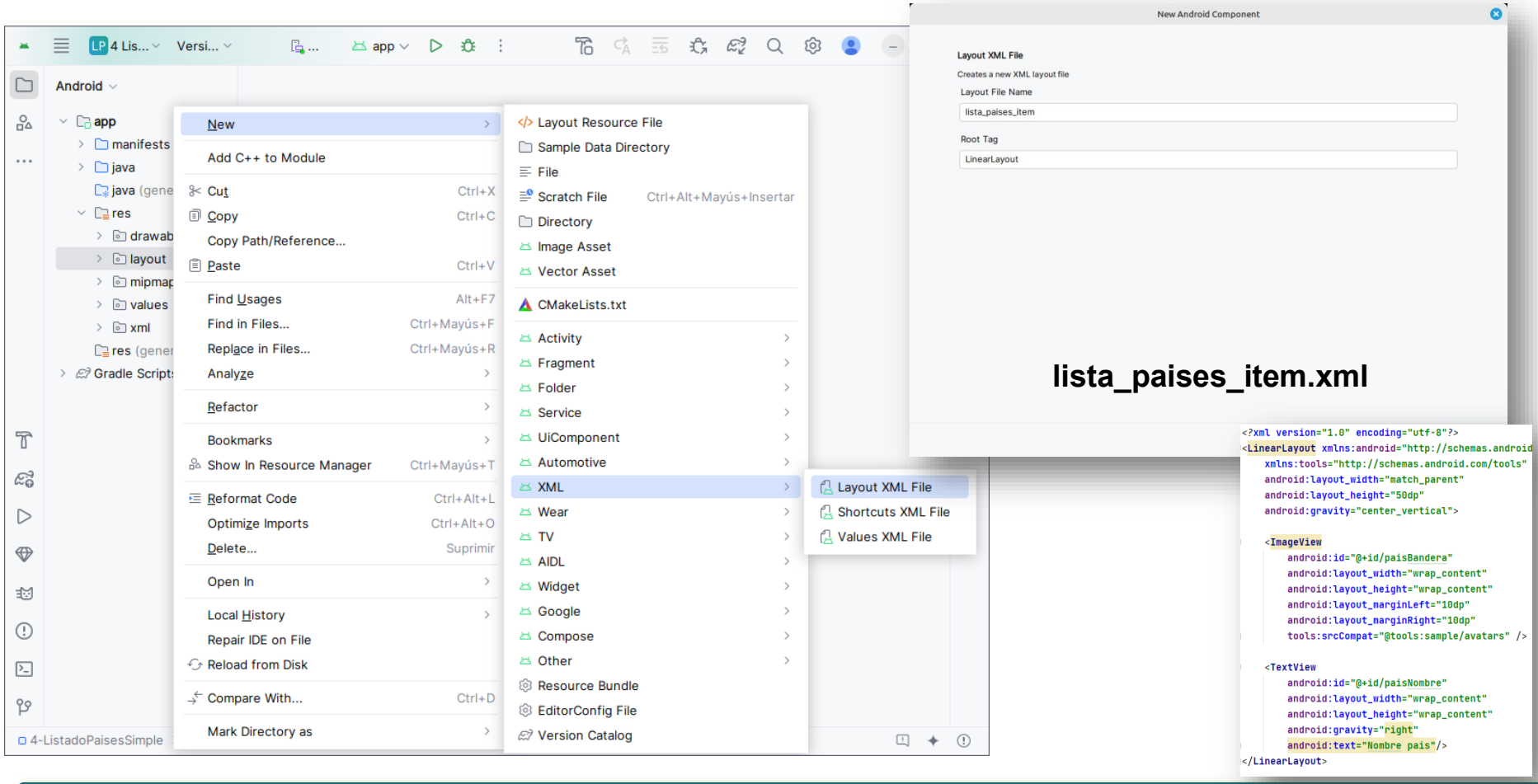




Método getView(): arquitectura básica



Crear el XML para el View de cada ítem



Diseño del View de cada ítem

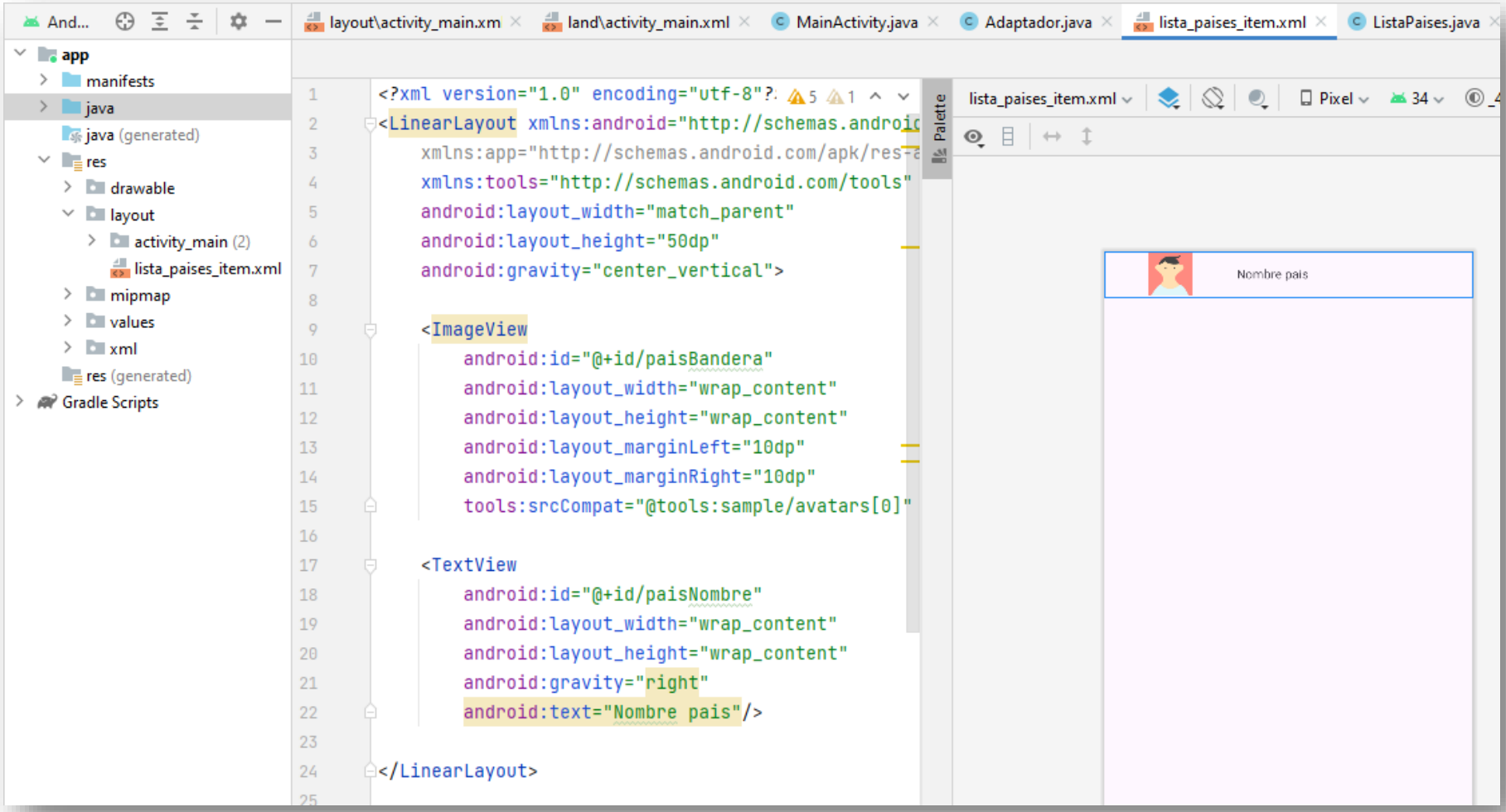


Imagen de cada ítem

- Se deben colocar en **`res/drawable`**
- Si en el `layout` no se define el tamaño (para que no haya reescalados), todas las imágenes deben ser del mismo tamaño y el adecuado a donde se van a mostrar. Por ejemplo 16x12
- Los formatos admitidos son PNG, JPG, WEBP y SVG
- Puede interesar que el fondo sea transparente (JPG no lo es)

Modelo (datos)

```
public class ListaPaises {
  private ArrayList<Pais> listaPaises;

  public ListaPaises () {
    listaPaises = new ArrayList<Pais>();
    listaPaises.add(new Pais("España",R.drawable.es));
    listaPaises.add(new Pais("Francia",R.drawable.fr));
    listaPaises.add(new Pais("Grecia",R.drawable.grecia));
    listaPaises.add(new Pais("Italia",R.drawable.it));
    listaPaises.add(new Pais("Reino unido",R.drawable.gb));
    ...
    }

  public ArrayList<Pais> getListaPaises() {
    return listaPaises;
   }
}
```

```
public class Pais {
    private String nombre;
    private int bandera;

    public Pais (String nombre,int bandera) {
        this.nombre=nombre;
        this.bandera=bandera;
    }

    public String getNombre(){
        return nombre;
    }

    public int getBandera() {
        return bandera;
    }

    public String toString() {
        return nombre;
    }
}
```

Controlador (Adaptador personalizado)

```
public class Adaptador extends ArrayAdapter<Pais> {

    private ArrayList<Pais> paises;
    private final LayoutInflater inflador;

    public Adaptador (Context contexto, ArrayList<Pais> paises) {
        super (contexto,0,paises); // Invocar al constructor de ArrayAdapter, pasando un 0 en el
        this.paises=paises;        //2º parámetro (el layout a usar) porque ahora vamos a usar el nuestro

        inflador = (LayoutInflater) contexto.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
    }

    @Override
    public View getView(int posicion, View vistaReciclada, ViewGroup padre ) {
        // ¿Se ha creado el view actual?
        if (vistaReciclada==null) {
            // Si es la primera vez, entonces crearlo
            vistaReciclada=inflador.inflate(R.layout.lista_paises_item,padre,false);
        }
        // Obtener las referencias de los elementos del view
        final TextView paisNombre = vistaReciclada.findViewById(R.id.paisNombre);
        final ImageView paisBandera = vistaReciclada.findViewById(R.id.paisBandera);

        // Obtener datos del país a meter en el View, sacándolos del ArrayList según la posición
        final Pais pais = getItem(posicion);          // Equivalente a paises.get(posicion);

        // Meter los datos en el view
        paisNombre.setText(pais.getNombre());
        paisBandera.setImageResource(pais.getBandera());

        return vistaReciclada;
    }
}
```

Código onCreate()

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


    // Instanciar objeto con el arrayList de objetos País:
    final ListaPaises datosPaises = new ListaPaises();

    // Instanciar un adaptador personalizado:
    final Adaptador adaptadorPaises = new Adaptador( this, datosPaises.getListaPaises() );

    // ArrayAdapter<Pais> adaptadorPaises = new ArrayAdapter<Pais>( this,
    //                                                  simple_list_item_checked,
    //                                                  datosPaises.getListaPaises() );

    // Obtener la referencia al ListView del layout de la actividad
    listView = findViewById(R.id.listView);

    // Asignar el adaptador al ListView
    listView.setAdapter(adaptadorPaises);

    // Establecer el manejador que se ejecutará al hacer clic en un ítem:
    listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView adapterView, View view, int posicion, long id) {
            // Codificar que sucederá con el ítem en el que se ha hecho clic

        }
}
```

Datos para el adaptador personalizado

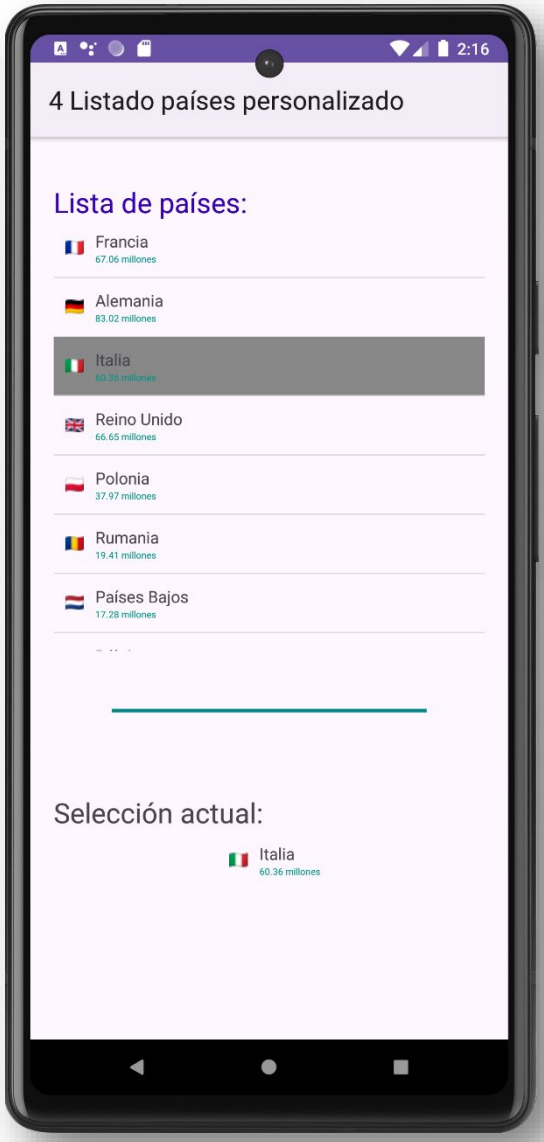
Si no se usara un adaptador personalizado se haría así. En este caso usa *toString()* en cada País para obtener su nombre

Tratamiento del elemento seleccionado

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView adapterView, View view,
                                        int posicion, long id) {
        // Suponiendo que es una selección simple.

        // Una forma es hacer un toString del ítem sacándolo del ListView
        // en la posición que se indica:

        String item=listView.getItemAtPosition(posicion).toString();

        // Otra es obtener el ítem del adaptador y usar los métodos
        // implementados en la clase País:

        String nombre = adaptadorPaises.getItem(posicion).getNombre();

        int bandera = adaptadorPaises.getItem(posicion).getBandera();

    }
});
```

Ejercicio 2 (1)

- Crea un proyecto nuevo con las siguientes características:
  - Nombre: “4 Listado países personalizado”
  - Paquete: dam.listadoPaisesPersonalizado
  - Directorio: 4-ListadoPaisesPersonalizado
- Cambia la presentación de cada ítem de la lista **para incluir la población** de cada país. Modifica:
  - El diseño del *View* de cada ítem
  - El modelo de datos
  - El método ***getView()*** del adaptador
- Cuando se selecciona un elemento, este debe cambiar su color de fondo y aparecer en la parte inferior con toda su información
- Organiza el código en paquetes:
  - Paquete Vista: clase del adaptador
  - Paquete Modelo: clases ListaPaises y Pais



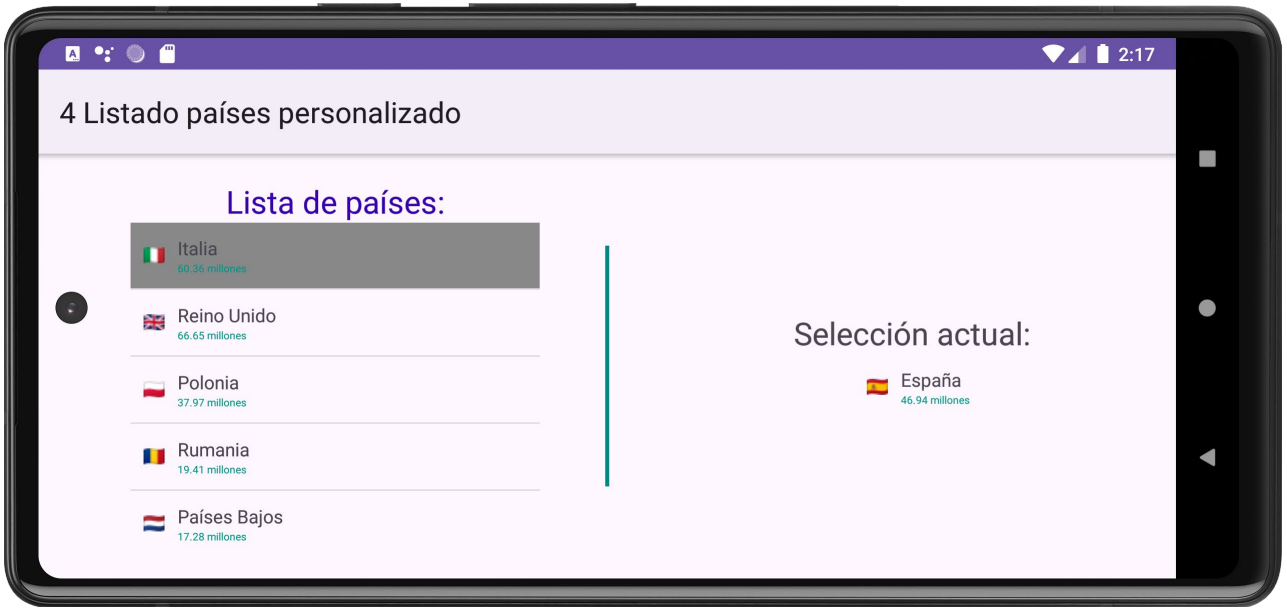
Ejercicio 2 (2)

- Como el diseño del país seleccionado ya está en otro fichero de diseño, para no copiar y pegar el html puedes usar un elemento include de XML:

```
<include layout="@layout/lista_paises_item" />
```

Ejercicio 2 (3)

- Al cambiar a la posición horizontal debe:
  - Recuperar el ítem seleccionado
  - Hacer *scroll* para que se coloque el primero



Ejercicio 2 (4)

- Se puede hacer que el ítem seleccionado cambie el color de fondo declarando el atributo `listSelector` en el XML del `ListView`. Pero solo cambia al interaccionar con el `ListView` en la GUI, no lo hace con `setItemChecked()`, y esto se necesita al cambiar de orientación
- Una forma de conseguirlo es:

1. En `onItemClick`() del manejador del `ListView`:
   - Guardar una referencia a la vista del ítem en el que se ha hecho clic
   - Cambiar el color de fondo a la vista del ítem en el que se ha hecho clic
   - Restaurar el color de fondo de la vista que anteriormente se hizo clic
2. En `getView`() del adaptador crear la vista con el color de fondo que le corresponda

```
If (listadoPaises.isItemChecked(posicion))          // Si es el que está seleccionado
   vistaReciclada.setBackgroundColor(colorFondo); // ponerle un color de fondo
else
   vistaReciclada.setBackgroundColor(Color.TRANSPARENT);  // Si no, quitárselo
```

Ejercicio 2 (5)

- En un cambio de orientación hay que guardar el nº del ítem seleccionado y recuperarlo, usando los métodos `onSaveInstanceState()` y `onRestoreInstanceState()`
- En `onRestoreInstanceState()` hay que usar los siguientes métodos de `ListView` para:

```
listView.setItemChecked (posición, true) // Marcar el ítem como seleccionado
listView.post(new Runnable() {
    @Override
    public void run() {
    listView.setSelection(posición); // Hacer scroll hasta él
    // Guardar referencia a la vista del seleccionado para usar en onItemClick()
    vistaUltimoItemSeleccionado = listView.getChildAt(posición -
                                  listView.getFirstVisiblePosition());
    }
});
```

Spinner con adaptador personalizado

- Al igual que un `ListView`, un `Spinner` puede usar un adaptador personalizado para mostrar elementos que no son un simple `String`
- La clase adaptador que se use debe implementar, además de `getView()`, el método `getDropDownView()`, que se ejecutará al pulsar el icono para desplegar la lista.
- Típicamente este método invoca a `getView()`

```
@Override
public View getDropDownView(int position, View convertView, ViewGroup parent) {
  return this.getView(position, convertView, parent);
}
```

Ejercicio 3 (1)

- A partir del proyecto “3 Registro” de la sesión anterior, crea un proyecto, con las siguientes características:
  - Nombre: “4 Registro”
  - Paquete: dam.registroV2
  - Directorio: 4-Registro
- Basándote en la aplicación de registro de la sesión anterior, implementa un adaptador personalizado para el `Spinner` de países. Ahora un país se definirá mediante un nombre y una bandera
- Organiza el código en los paquetes indicados en el ejercicio anterior
- Pon el mismo icono a la aplicación que el usado en la aplicación de registro de la sesión anterior

Ejercicio 3 (2)