

Curso Android V3.0



ANDROID

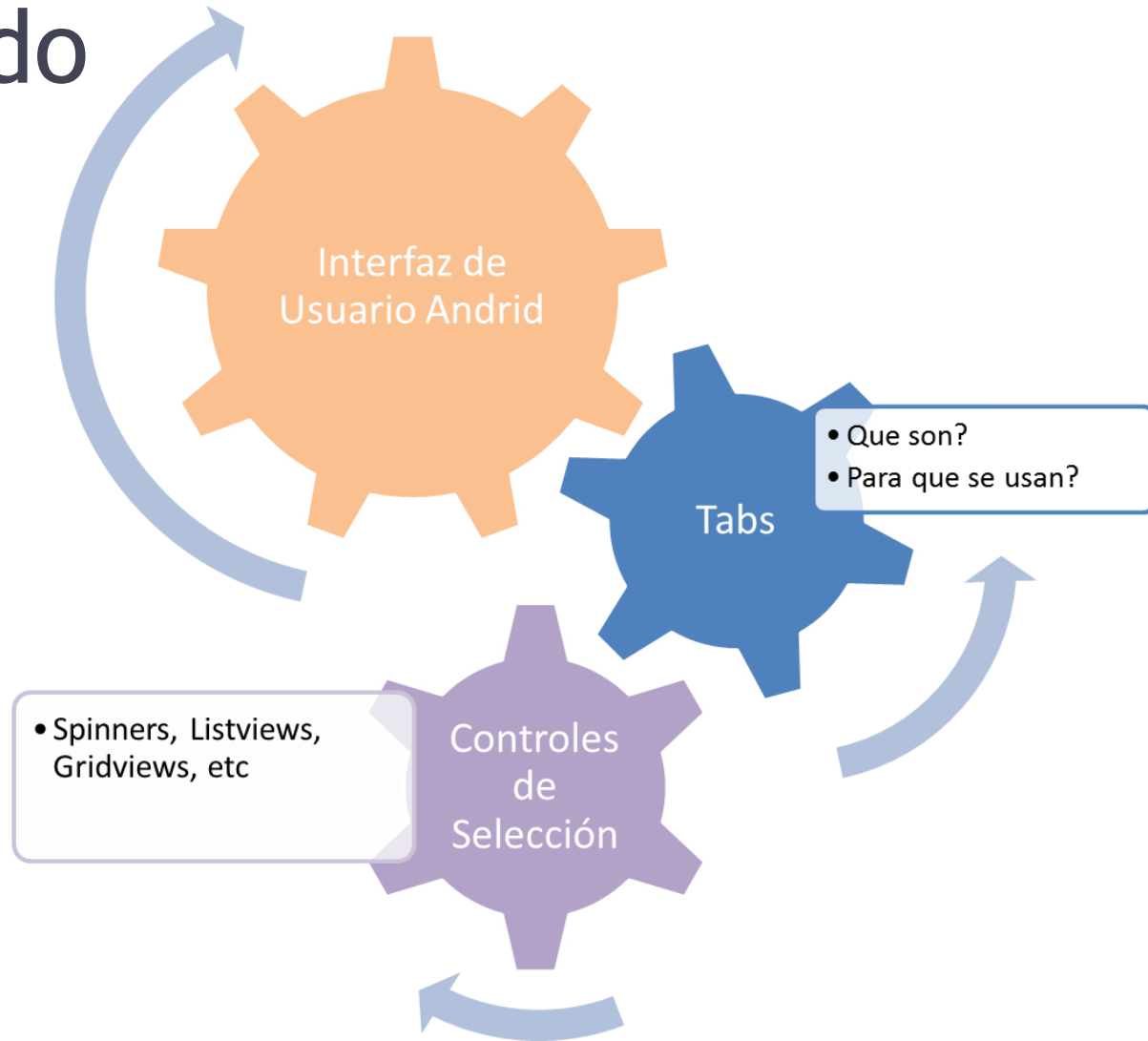
Listas y Tabs

Emiliano Gonzalez

Pedro Coronel

2013

Contenido



Continuación de Intents

Boton Volver

```
<Button  
    android:layout_width="fill_parent"  
    android:layout_height="wrap_content"  
    android:onClick="volver"  
    android:text="Volver" />
```

```
public void volver(View view)  
{  
    finish();  
}
```

Enviar datos entre Intents

```
public void onClick(View view){  
    Intent otra = new Intent(this, OtraActivity.class);  
    otra.putExtra("valor","hola");  
    startActivity(otra);  
}
```

En el OnCreate de OtraActivity

```
Intent intent = getIntent();  
TextView tw = (TextView)findViewById(R.id.tw_otra);  
tw.setText(intent.getStringExtra("valor"));
```

URL Intent

- En el manifest

- `<uses-permission
android:name="android.permission.INTERNET"/>`

- En la clase java

- `intent = new Intent(Intent.ACTION_VIEW,`
 - `Uri.parse("http://www.ipositivo.com"));`

Call Intent

- En el manifiest

- `<uses-permission
 android:name="android.permission.CALL_PHONE" />`

- En la clase java

- `intent = new Intent(Intent.ACTION_CALL,`
 - `Uri.parse("tel:(+595)971507080"));`

Controles de Selección

- Adapters
- Control Spinner
- ListView
- GridView

Adaptadores en Android (*adapters*)


- Un adaptador representa algo así como una interfaz común al modelo de datos que existe por detrás de todos los controles de selección. Dicho de otra forma, todos los controles de selección accederán a los datos que contienen a través de un adaptador.
- Además de proveer de datos a los controles visuales, el adaptador también será responsable de generar a partir de estos datos las vistas específicas que se mostrarán dentro del control de selección

Tipos de Adapters

- ArrayAdapter. Es el más sencillo de todos los adaptadores, y provee de datos a un control de selección a partir de un array de objetos de cualquier tipo.
- SimpleAdapter. Se utiliza para mapear datos sobre los diferentes controles definidos en un fichero XML.
- SimpleCursorAdapter. Se utiliza para mapear las columnas de un cursor sobre los diferentes elementos visuales contenidos en el control de selección.

Ejemplo

```
1  final String[] datos =  
2      new String[]{"Elem1", "Elem2", "Elem3", "Elem4", "Elem5"};  
3  
4  ArrayAdapter<String> adaptador =  
5      new ArrayAdapter<String>(this,  
6          android.R.layout.simple_spinner_item, datos);
```



Control Spinner

- Las listas desplegadas en Android se llaman Spinner. Funcionan de forma similar al de cualquier control de este tipo, el usuario selecciona la lista, se muestra una especie de lista emergente al usuario con todas las opciones disponibles y al seleccionarse una de ellas ésta queda fijada en el control.


Ejemplo

```
1 <Spinner android:id="@+id/CmbOpciones"  
2     android:layout_width="fill_parent"  
3     android:layout_height="wrap_content" />
```

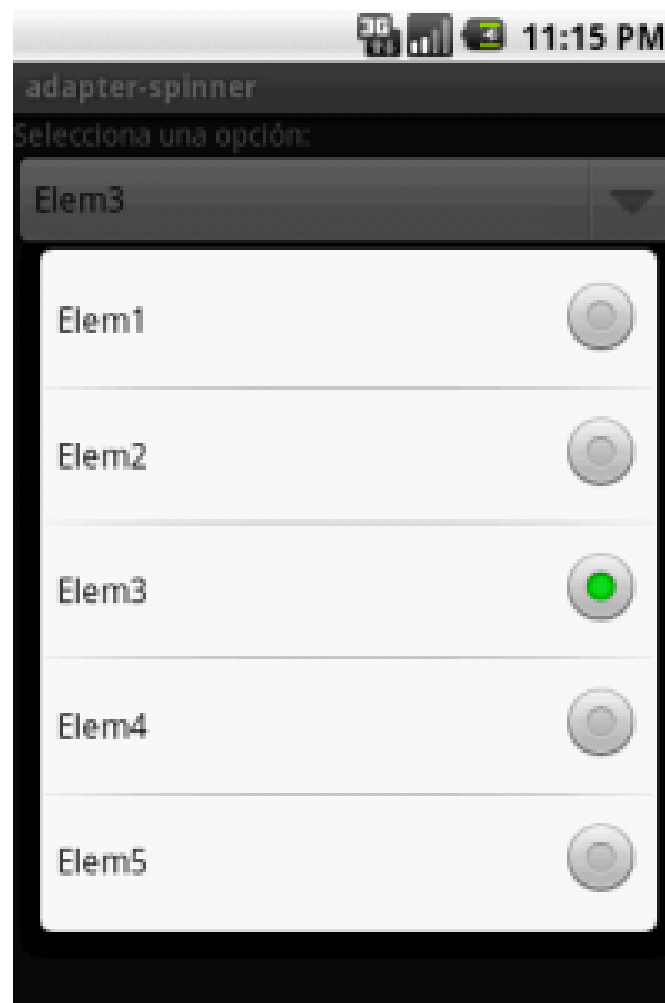
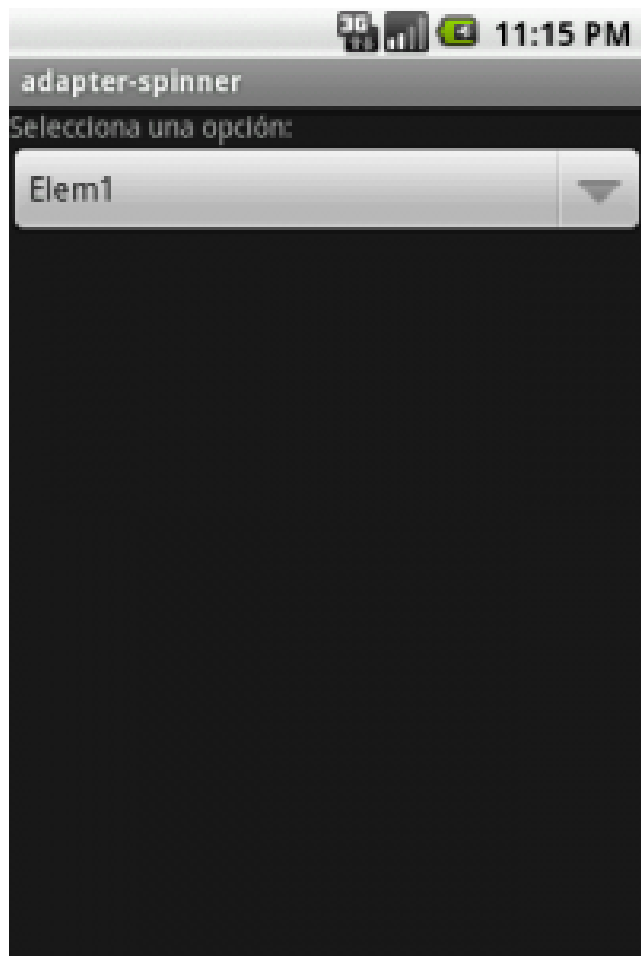
```
1 final Spinner cmbOpciones = (Spinner) findViewById(R.id.CmbOpciones) ;  
2  
3 adaptador.setDropDownViewResource(  
4     android.R.layout.simple_spinner_dropdown_item) ;  
5  
6 cmbOpciones.setAdapter(adaptador) ;
```

Ejemplo

```
1  cmbOpciones.setOnItemSelectedListener(  
2      new AdapterView.OnItemClickListener() {  
3          public void onItemClick(AdapterView<?> parent,  
4              android.view.View v, int position, long id) {  
5              lblMensaje.setText("Seleccionado: " + datos[position]);  
6          }  
7  
8          public void onNothingSelected(AdapterView<?> parent) {  
9              lblMensaje.setText("");  
10         }  
11     });
```



Resultado



ListView

- Un control ListView muestra al usuario una lista de opciones seleccionables directamente sobre el propio control, sin listas emergentes como en el caso del control Spinner. En caso de existir más opciones de las que se pueden mostrar sobre el control se podrá por supuesto hacer scroll sobre la lista para acceder al resto de elementos.

Ejemplo

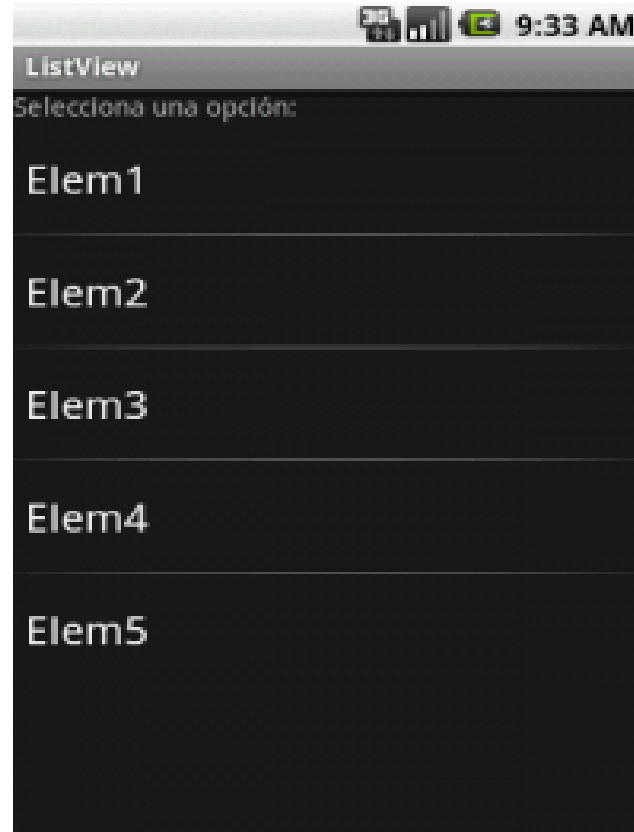
```
1 <ListView android:id="@+id/LstOpciones"
2         android:layout_width="wrap_content"
3         android:layout_height="wrap_content" />
```

```
1 final String[] datos =
2     new String[]{"Elem1", "Elem2", "Elem3", "Elem4", "Elem5"};
3
4 ArrayAdapter<String> adaptador =
5     new ArrayAdapter<String>(this,
6         android.R.layout.simple_list_item_1, datos);
7
8 ListView lstOpciones = (ListView) findViewById(R.id.LstOpciones);
9
10 lstOpciones.setAdapter(adaptador);
```

Ejemplo

```
1 lstOpciones.setOnItemClickListener(new OnItemClickListener() {  
2     @Override  
3     public void onItemClick(AdapterView<?> a, View v, int position, long id) {  
4         //Acciones necesarias al hacer click  
5     }  
6 });
```

Resultado:



GridView

- El control GridView de Android presenta al usuario un conjunto de opciones seleccionables distribuidas de forma tabular, o dicho de otra forma, divididas en filas y columnas

Ejemplo

```
1 <GridView android:id="@+id/GridOpciones"  
2     android:layout_width="fill_parent"  
3     android:layout_height="fill_parent"  
4     android:numColumns="auto_fit"  
5     android:columnWidth="80px"  
6     android:horizontalSpacing="5px"  
7     android:verticalSpacing="10px"  
8     android:stretchMode="columnWidth" />
```



Ejemplo

```
1  private String[] datos = new String[25];
2  //...
3  for(int i=1; i<=25; i++)
4      datos[i-1] = "Dato " + i;
5
6  ArrayAdapter<String> adaptador =
7      new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, datos);
8
9  final GridView grdOpciones = (GridView) findViewById(R.id.GridOpciones);
10
11  grdOpciones.setAdapter(adaptador);
```

Ejemplo

```
1  grdOpciones.setOnItemClickListener(  
2      new AdapterView.OnItemClickListener() {  
3          public void onItemClick(AdapterView<?> parent,  
4              android.view.View v, int position, long id) {  
5              lblMensaje.setText("Seleccionado: " + datos[position]);  
6          }  
7  
8          public void onNothingSelected(AdapterView<?> parent) {  
9              lblMensaje.setText("");  
10         }  
11     });
```

Resultado:



Tab Layout

- En Android, el elemento principal de un conjunto de pestañas será el control `TabHost`.
- Dentro de éste vamos a incluir un `LinearLayout` que nos servirá para distribuir verticalmente las secciones principales del layout: la sección de pestañas en la parte superior y la sección de contenido en la parte inferior.
- La sección de pestañas se representará mediante un elemento `TabWidget`.
- Como contenedor para el contenido de las pestañas añadiremos un `FrameLayout`.

TabHost [id/tabhost]

LinearLayout

TabWidget [id/tabs]

FrameLayout [id/tabcontent]

LinearLayout [id/tab1]

LinearLayout [id/tab2]

...

Layout:

```
1  <TabHost android:id="@android:id/tabhost"
2      android:layout_width="match_parent"
3      android:layout_height="match_parent">
4
5      <LinearLayout
6          android:orientation="vertical"
7          android:layout_width="fill_parent"
8          android:layout_height="fill_parent" >
9
10         <TabWidget android:layout_width="match_parent"
11             android:layout_height="wrap_content"
12             android:id="@android:id/tabs" />
13
14         <FrameLayout android:layout_width="match_parent"
15             android:layout_height="match_parent"
16             android:id="@android:id/tabcontent" >
17
18             <LinearLayout android:id="@+id/tab1"
19                 android:orientation="vertical"
20                 android:layout_width="match_parent"
21                 android:layout_height="match_parent" >
22                 <TextView android:id="@+id/textView1"
23                     android:text="Contenido Tab 1"
24                     android:layout_width="wrap_content"
25                     android:layout_height="wrap_content" />
26             </LinearLayout>
```

```
28         <LinearLayout android:id="@+id/tab2"
29             android:orientation="vertical"
30             android:layout_width="match_parent"
31             android:layout_height="match_parent" >
32             <TextView android:id="@+id/textView2"
33                 android:text="Contenido Tab 2"
34                 android:layout_width="wrap_content"
35                 android:layout_height="wrap_content" />
36         </LinearLayout>
37
38         </FrameLayout>
39     </LinearLayout>
40
41 </TabHost>
```

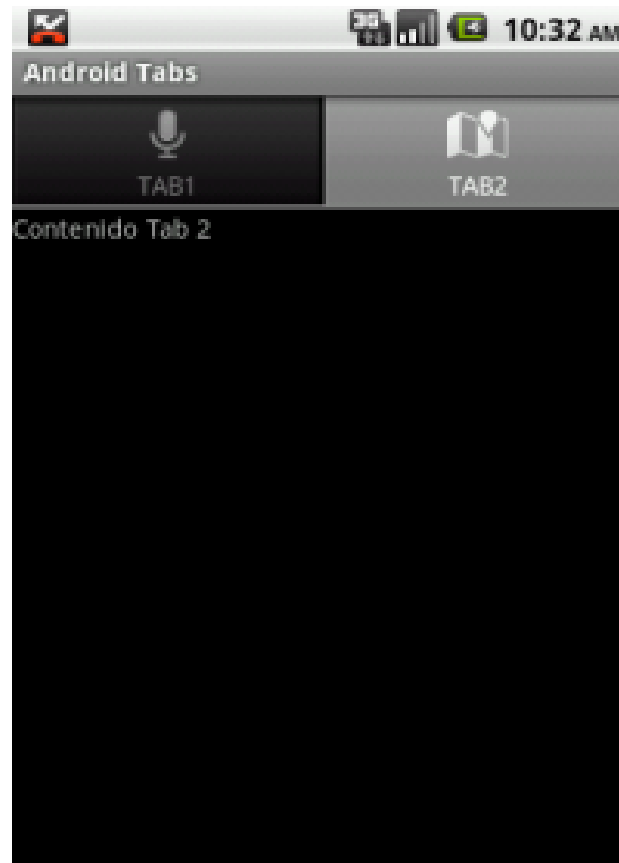
Source

```
1  Resources res = getResources();
2
3  TabHost tabs=(TabHost)findViewById(android.R.id.tabhost);
4  tabs.setup();
5
6  TabHost.TabSpec spec=tabs.newTabSpec("mitab1");
7  spec.setContent(R.id.tab1);
8  spec.setIndicator("TAB1",
9      res.getDrawable(android.R.drawable.ic_btn_speak_now));
10 tabs.addTab(spec);
11
12 spec=tabs.newTabSpec("mitab2");
13 spec.setContent(R.id.tab2);
14 spec.setIndicator("TAB2",
15     res.getDrawable(android.R.drawable.ic_dialog_map));
16 tabs.addTab(spec);
17
18 tabs.setCurrentTab(0);
```

Source

```
1  tabs.setOnTabChangeListener(new OnTabChangeListener() {  
2      @Override  
3      public void onTabChanged(String tabId) {  
4          Log.i("AndroidTabsDemo", "Pulsada pestaña: " + tabId);  
5      }  
6  });
```

Resultado



Ejercicios

ListView Complejo


- Definir un ListView en el main layout
- Crear una clase Titular que contenga los atributos título y subtítulo.
- Crear un layout que represente a la clase titular
- Crear un ArrayAdapter para manejar la clase Titular
- Poblar un ListView con el ArrayAdapter

Layout Principal

```
1  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      android:orientation="vertical"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent" >
5
6      <TextView android:id="@+id/LblEtiqueta"
7          android:layout_width="match_parent"
8          android:layout_height="wrap_content"
9          android:text="@string/selecciona_una_opcion" />
10
11     <ListView android:id="@+id/LstOpciones"
12         android:layout_width="match_parent"
13         android:layout_height="wrap_content" />
14
15 </LinearLayout>
```

Clase Titular

```
1  package py.com.cursoandroid;
2
3  public class Titular
4  {
5      private String titulo;
6      private String subtitulo;
7
8      public Titular(String tit, String sub){
9          titulo = tit;
10         subtitulo = sub;
11     }
12
13     public String getTitulo(){
14         return titulo;
15     }
16
17     public String getSubtitulo(){
18         return subtitulo;
19     }
20 }
```



Layout del listitem_titular.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <LinearLayout
4      xmlns:android="http://schemas.android.com/apk/res/android"
5      android:layout_width="wrap_content"
6      android:layout_height="wrap_content"
7      android:orientation="vertical">
8
9      <TextView android:id="@+id/LblTitulo"
10         android:layout_width="fill_parent"
11         android:layout_height="wrap_content"
12         android:textStyle="bold"
13         android:textSize="20px" />
14
15      <TextView android:id="@+id/LblSubTitulo"
16         android:layout_width="fill_parent"
17         android:layout_height="wrap_content"
18         android:textStyle="normal"
19         android:textSize="12px" />
20
21  </LinearLayout>
```

Declarar el array de Titulares

- `private Titular[] datos = new Titular[25];`

ArrayAdapter para Titulares

```
1  class AdaptadorTitulares extends ArrayAdapter {
2
3      Activity context;
4
5      AdaptadorTitulares(Activity context) {
6          super(context, R.layout.listitem_titular, datos);
7          this.context = context;
8      }
9
10     public View getView(int position, View convertView, ViewGroup parent) {
11         LayoutInflater inflater = context.getLayoutInflater();
12         View item = inflater.inflate(R.layout.listitem_titular, null);
13
14         TextView lblTitulo = (TextView)item.findViewById(R.id.LblTitulo);
15         lblTitulo.setText(datos[position].getTitulo());
16
17         TextView lblSubtitulo = (TextView)item.findViewById(R.id.LblSubTitulo);
18         lblSubtitulo.setText(datos[position].getSubtitulo());
19
20         return(item);
21     }
22 }
```

Clase Main

```
1  datos =
2      new Titular[]{
3          new Titular("Título 1", "Subtítulo largo 1"),
4          new Titular("Título 2", "Subtítulo largo 2"),
5          new Titular("Título 3", "Subtítulo largo 3"),
6          new Titular("Título 4", "Subtítulo largo 4"),
7          new Titular("Título 5", "Subtítulo largo 5")};
8
9  //...
10 //...
11
12 AdaptadorTitulares adaptador =
13     new AdaptadorTitulares(this);
14
15 ListView lstOpciones = (ListView)findViewById(R.id.LstOpciones);
16
17 lstOpciones.setAdapter(adaptador);
```

TAREA

- Agregar al Bloc de Notas una lista de elementos utilizando el Layout con un Título y parte del texto del contenido de la nota. Cada uno de los elementos debe mostrar el color correspondiente a su Prioridad (Rojo, verde o Amarillo) según corresponda.