

Ventás de Diálogos

Sumario

Introdución

Caso práctico

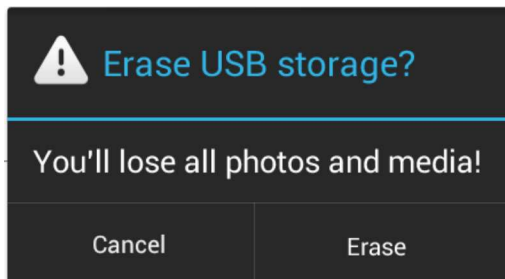
- O XML do Layout
- Definición de recursos tipo array en XML
- Definición do recurso de tipo Layout usado polo diálogo con entrada de texto (o último)
- Código Java
- Ventás de diálogo personalizadas

DialogFragment

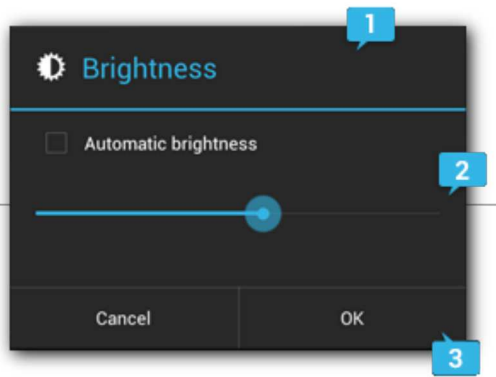
- Caso práctico
 - O XML do DialogFragment
 - O código Java do DialogFragment
 - O XML da Activity
 - O código Java da Activity
- Unha variación

Introdución

- Un **diálogo** é unha ventá pequena que lle pregunta ao usuario para tomar unha decisión, para informalo, ou para que o usuario introduza información.
- Non cubre toda a pantalla
- É usado para que o usuario tome algún tipo de acción antes de seguir coa aplicación.



- Hai catro tipos de ventás de diálogo:
 - AlertDialog**: pode conter de cero a tres botóns, unha lista, RadioButtons, CheckBoxes, etc.
 - ProgressDialog**: amosa unha barra de progreso. Herda de da clase AlertDialog. Verase na Unidade 5.
 - DatePickerDialog**: este diálogo é para seleccionar unha fecha.
 - TimePickerDialog**: este diálogo permite seleccionar unha hora.
- Neste unidade imos centrarnos nos **AlertDialog**
 - Un diálogo baseado na clase AlertDialog ten 3 rexións:

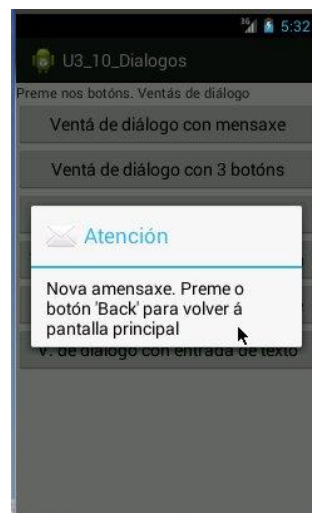
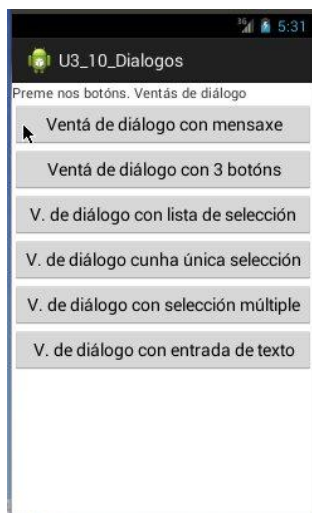


- **1.-Título:** é opcional.
- **2.-Área de contido:** pode amosar unha mensaxe, lista, un layout personalizado, etc.
- **3.-Botóns de acción:** Non debe haber máis de tres botóns.
- Hai tres tipos de botóns:
 - **Positivo:** usárase para aceptar ou continuar unha acción.
 - **Negativo:** usárase cando se desexe cancelar unha acción.
 - **Neutral:** usárase cando non se sabe que facer coa acción.
- Para amosar un diálogo úsase o método **showDialog()** indicando un parámetro enteiro, único na actividade
- Iremos construír os diálogos no método **onCreateDialog()**, que recibe un enteiro facendo referencia ao diálogo e devolve un obxecto de tipo diálogo.
- **Referencias:**
 - <http://developer.android.com/design/building-blocks/dialogs.html>
 - <http://developer.android.com/guide/topics/ui/dialogs.html>

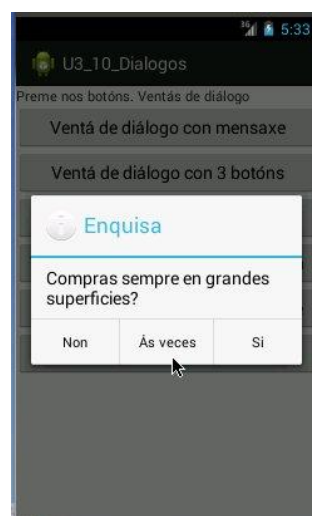
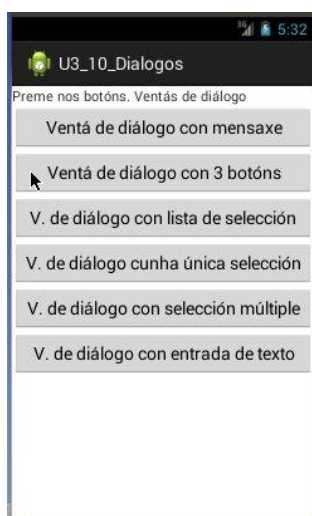
Caso práctico

- Creamos un novo proxecto: **U3_10_Dialogos**
- Esta aplicación está baseada no curso de Android da Aula Mentor: <http://www.mentor.mec.es/>
- As seguintes imaxes amosan o funcionamento da aplicación.
- Todos os elementos dos diálogos teñen asociado un Toast como acción, aínda que só se amose na última imaxe da seguinte secuencia.

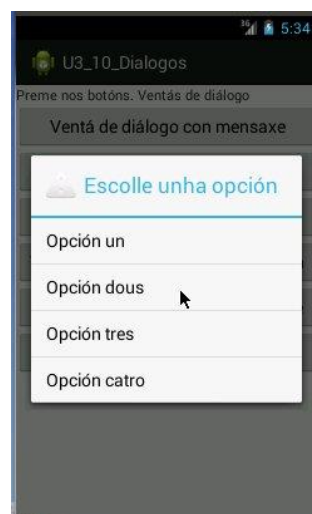
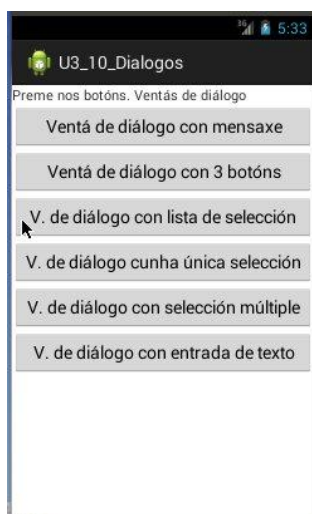
Aplicación Diálogos



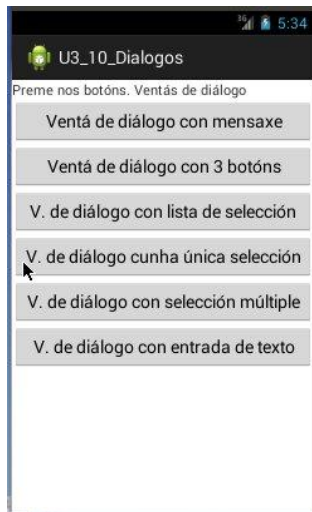
Diálogo informativo.



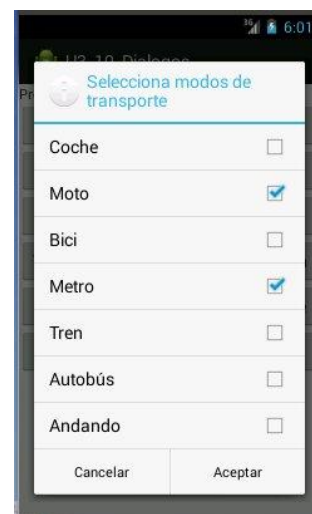
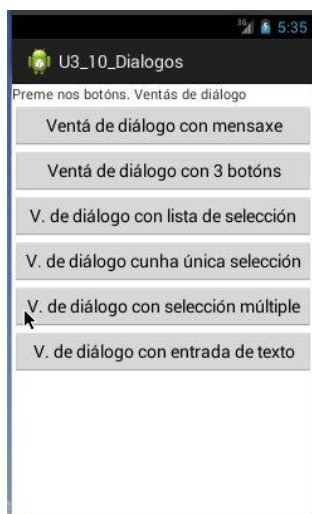
Diálogo con botóns, pode ter tamén 2 ou 1 botón soamente.



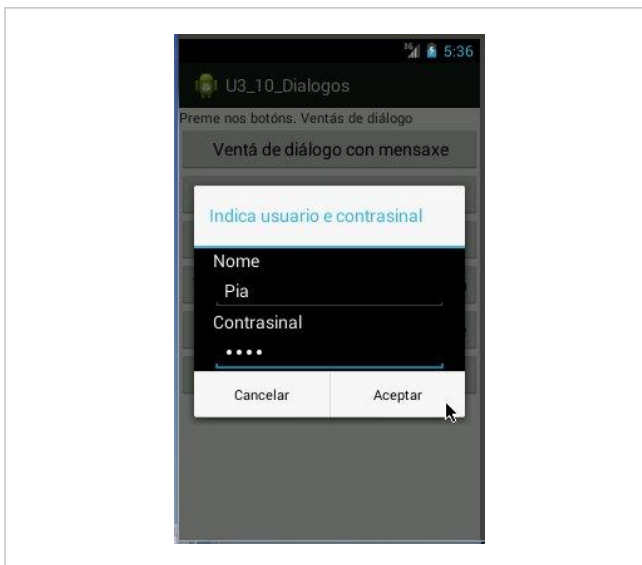
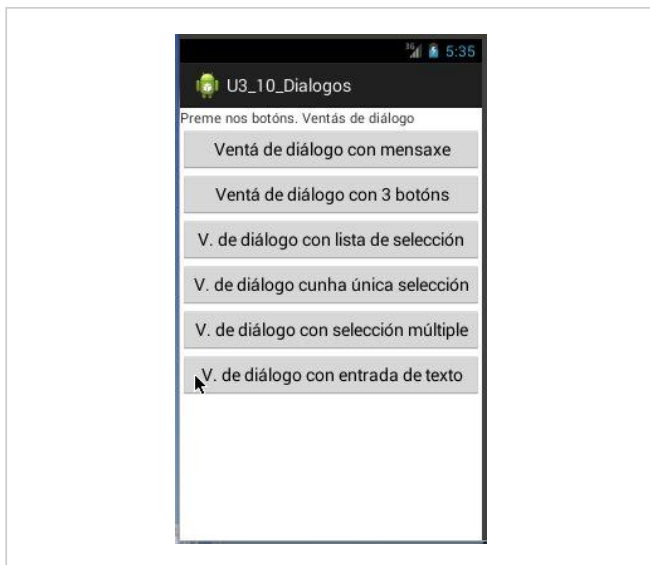
Lista de selección (Spinner)



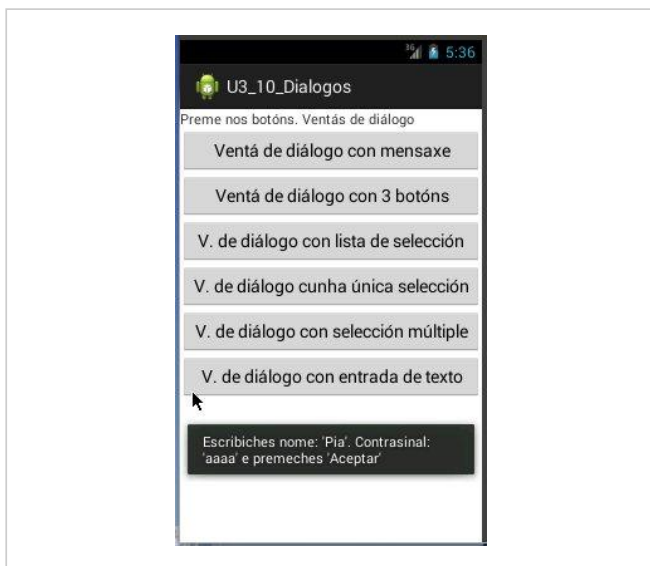
RadioButtons



Múltiple selección con algúns ítems marcados por defecto.



Layout definido polo usuario para a entrada de información a través dun cadro de diálogo.



O XML do Layout

- Todos Botóns executan o mesmo método para o evento Click.

```

1 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2   android:layout_width="fill_parent"
3   android:layout_height="fill_parent"
4   android:orientation="vertical" >
5
6   <TextView
7     android:layout_width="match_parent"
8     android:layout_height="wrap_content"
9     android:text="Preme nos botóns. Ventás de diálogo" />
10
11   <ScrollView
12     android:id="@+id/screen"
13     android:layout_width="match_parent"
14     android:layout_height="match_parent"
15     android:orientation="vertical" >
16
17     <LinearLayout
18       android:layout_width="match_parent"
19       android:layout_height="match_parent"
20       android:orientation="vertical" >

```

```

21
22     <Button
23         android:id="@+id/btn_dialogo"
24         android:layout_width="match_parent"
25         android:layout_height="wrap_content"
26         android:onClick="onBotonClick"
27         android:text="Ventá de diálogo con mensaxe" >
28     </Button>
29
30     <Button
31         android:id="@+id/btn_diag_tres_botons"
32         android:layout_width="match_parent"
33         android:layout_height="wrap_content"
34         android:onClick="onBotonClick"
35         android:text="Ventá de diálogo con 3 botóns" />
36
37     <Button
38         android:id="@+id/btn_diag_list_selecc"
39         android:layout_width="match_parent"
40         android:layout_height="wrap_content"
41         android:onClick="onBotonClick"
42         android:text="V. de diálogo con lista de selección" />
43
44     <Button
45         android:id="@+id/btn_diag_radio_button"
46         android:layout_width="match_parent"
47         android:layout_height="wrap_content"
48         android:onClick="onBotonClick"
49         android:text="V. de diálogo cunha única selección" />
50
51     <Button
52         android:id="@+id/btn_diag_checkbox"
53         android:layout_width="match_parent"
54         android:layout_height="wrap_content"
55         android:onClick="onBotonClick"
56         android:text="V. de diálogo con selección múltiple" />
57
58     <Button
59         android:id="@+id/btn_diag_entrada_texto"
60         android:layout_width="match_parent"
61         android:layout_height="wrap_content"
62         android:onClick="onBotonClick"
63         android:text="V. de diálogo con entrada de texto" />
64 </LinearLayout>
65 </ScrollView>
66
67 </LinearLayout>

```

Definición de recursos tipo array en XML

- Aproveitamos o ficheiro **/res/values/strings.xml** e definimos dous recursos de tipo Array, para a lista de selección e para a selección múltiple.

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3
4     <string name="app_name">U3_10_Dialogos</string>
5     <string name="action_settings">Settings</string>
6
7
8     <string-array name="elementos_dialog_seleccion">
9         <item>Opción un</item>
10        <item>Opción dous</item>
11        <item>Opción tres</item>
12        <item>Opción catro</item>
13    </string-array>
14
15    <string-array name="elementos_dialog_seleccion2">
16        <item>Coche</item>
17        <item>Moto</item>
18        <item>Bici</item>
19        <item>Metro</item>
20        <item>Tren</item>
21        <item>Autobús</item>
22        <item>Andando</item>
23    </string-array>
24
25 </resources>

```

Definición do recurso de tipo Layout usado polo diálogo con entrada de texto (o último)

Explicado posteriormente.

■ Ficheiro `/res/layout/dialogo_entrada_texto.xml`

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="match_parent"
4     android:layout_height="wrap_content"
5     android:background="#000"
6     android:orientation="vertical">
7
8     <TextView
9         android:layout_height="wrap_content"
10        android:layout_width="wrap_content"
11        android:layout_marginLeft="20dp"
12        android:layout_marginRight="20dp"
13        android:text="Nome"
14        android:gravity="left"
15        android:textAppearance="?android:attr/textAppearanceMedium" />
16
17     <EditText
18         android:id="@+id/et_nome"
19         android:layout_height="wrap_content"
20         android:layout_width="match_parent"
21         android:layout_marginLeft="20dp"
22         android:layout_marginRight="20dp"
23         android:scrollHorizontally="true"
24         android:autoText="false"
25         android:capitalize="none"
26         android:gravity="fill_horizontal"
27         android:textAppearance="?android:attr/textAppearanceMedium" />
28
29     <TextView
30         android:layout_height="wrap_content"
31         android:layout_width="wrap_content"
32         android:layout_marginLeft="20dp"
33         android:layout_marginRight="20dp"
34         android:text="Contrasinal"
35         android:gravity="left"
36         android:textAppearance="?android:attr/textAppearanceMedium" />
37
38     <EditText
39         android:id="@+id/et_contrasinal"
40         android:layout_height="wrap_content"
41         android:layout_width="match_parent"
42         android:layout_marginLeft="20dp"
43         android:layout_marginRight="20dp"
44         android:scrollHorizontally="true"
45         android:gravity="fill_horizontal"
46         android:password="true"
47         android:textAppearance="?android:attr/textAppearanceMedium" />
48
49 </LinearLayout>

```

Código Java

```

1 package com.example.u3_10_dialogos;
2
3 import android.app.Activity;
4 import android.app.AlertDialog;
5 import android.app.Dialog;
6 import android.app.ProgressDialog;
7 import android.content.Context;
8 import android.content.DialogInterface;
9 import android.content.res.Resources;
10 import android.os.Bundle;
11 import android.os.Handler;
12 import android.os.Message;
13 import android.view.LayoutInflater;
14 import android.view.Menu;
15 import android.view.View;
16 import android.widget.TextView;
17 import android.widget.Toast;
18
19 public class U3_10_Dialogos extends Activity {
20     private static final int DIALOGO_MENSAXE = 1;
21     private static final int DIALOGO_TRES_BOTONS = 2;
22     private static final int DIALOGO_LISTA = 3;
23     private static final int DIALOGO_RADIO_BUTTON = 4;
24     private static final int DIALOGO_CHECK_BOX = 5;

```

```

25 private static final int DIALOGO_ENTRADA_TEXTO = 6;
26
27 // Variable para crear as ventás de diálogo
28 AlertDialog.Builder venta;
29
30 @Override
31 protected void onCreate(Bundle savedInstanceState) {
32     super.onCreate(savedInstanceState);
33     setContentView(R.layout.activity_u3_10_dialogos);
34 }
35
36
37 protected Dialog onCreateDialog(int id) {
38     switch (id) {
39
40         case DIALOGO_MENSAXE:
41             venta = new AlertDialog.Builder(this);
42             venta.setTitle("Atención");
43             venta.setMessage("Nova mensaxe. Preme o botón 'Back' para volver á pantalla principal");
44             venta.setIcon(android.R.drawable.ic_dialog_email);
45             return venta.create();
46
47
48
49         case DIALOGO_TRES_BOTONS:
50             venta = new AlertDialog.Builder(this);
51             venta.setIcon(android.R.drawable.ic_dialog_info);
52             venta.setTitle("Enquisa");
53             venta.setMessage("Compras sempre en grandes superficies?");
54             venta.setCancelable(false);
55             venta.setPositiveButton("Si", new DialogInterface.OnClickListener() {
56                 public void onClick(DialogInterface dialog, int boton) {
57                     /* Sentencias se o usuario preme Si */
58                     Toast.makeText(getApplicationContext(), "Premeches 'Si'", 1).show();
59                 }
60             });
61             venta.setNegativeButton("Non", new DialogInterface.OnClickListener() {
62                 public void onClick(DialogInterface dialog, int boton) {
63                     /* Sentencias se o usuario preme Non */
64                     Toast.makeText(getApplicationContext(), "Premeches 'Non'", 1).show();
65                 }
66             });
67             venta.setNeutralButton("Ás veces", new DialogInterface.OnClickListener() {
68                 public void onClick(DialogInterface dialog, int boton) {
69                     /* Sentencias se o usuario preme Ás veces */
70                     Toast.makeText(getApplicationContext(), "Premeches 'Ás veces'", 1).show();
71                 }
72             });
73             return venta.create();
74
75
76
77         case DIALOGO_LISTA:
78             venta = new AlertDialog.Builder(this);
79             venta.setIcon(android.R.drawable.ic_dialog_alert);
80             venta.setTitle("Escolle unha opción");
81             venta.setItems(R.array.elementos_dialog_seleccion, new DialogInterface.OnClickListener() {
82                 public void onClick(DialogInterface dialog, int opcion) {
83                     // O usuario selecciona unha das opcións do listado
84                     String[] opciones = getResources().getStringArray(R.array.elementos_dialog_seleccion);
85                     Toast.makeText(getApplicationContext(), "Seleccionaches: " + opciones[opcion] + "", 1).show();
86                 }
87             });
88             return venta.create();
89
90
91
92         case DIALOGO_RADIO_BUTTON:
93             venta = new AlertDialog.Builder(this);
94             venta.setIcon(android.R.drawable.ic_dialog_info);
95             venta.setTitle("Selecciona un smartpohone");
96             // Non incluír mensaxe dentro de este tipo de diálogo!!!
97             final CharSequence[] smartphones = { "iPhone", "Blackberry", "Android" };
98             venta.setSingleChoiceItems(smartphones, 0, new DialogInterface.OnClickListener() {
99                 public void onClick(DialogInterface dialog, int opcion) {
100                     // Evento que ocorre cando o usuario selecciona una opción
101                     Toast.makeText(getApplicationContext(), "Seleccionaches: " + smartphones[opcion], Toast.LENGTH_SHORT).show();
102                 }
103             });
104             venta.setPositiveButton("Aceptar", new DialogInterface.OnClickListener() {
105                 public void onClick(DialogInterface dialog, int boton) {
106                     Toast.makeText(getApplicationContext(), "Premeches 'Aceptar'", 1).show();
107                 }
108             });
109             venta.setNegativeButton("Cancelar", new DialogInterface.OnClickListener() {
110                 public void onClick(DialogInterface dialog, int boton) {
111                     Toast.makeText(getApplicationContext(), "Premeches 'Cancelar'", 1).show();
112                 }
113             });
114             return venta.create();
115
116
117
118     }
119 }

```



```

113     });
114     return venta.create();
115
116
117
118     case DIALOGO_CHECK_BOX:
119         venta = new AlertDialog.Builder(this);
120         venta.setIcon(android.R.drawable.ic_dialog_info);
121         venta.setTitle("Selecciona modos de transporte");
122         Resources res = getResources();
123         final String[] matriz = res.getStringArray(R.array.elementos_dialog_seleccion2);
124         // Non incluír mensaxe dentro de este tipo de diálogo!!!
125         venta.setMultiChoiceItems(matriz, new boolean[] { false, true, false, true, false, false, false }, new
DialogInterface.OnMultiChoiceClickListener() {
126             public void onClick(DialogInterface dialog, int opcion, boolean isChecked) {
127                 // Evento que ocorre cando o usuario selecciona unha opción
128                 if (isChecked)
129                     Toast.makeText(getApplicationContext(), "Seleccionaches " + matriz[opcion], Toast.LENGTH_SHORT).show();
130                 else
131                     Toast.makeText(getApplicationContext(), "Deseleccionaches " + matriz[opcion], Toast.LENGTH_SHORT).show();
132             }
133         });
134         venta.setPositiveButton("Aceptar", new DialogInterface.OnClickListener() {
135             public void onClick(DialogInterface dialog, int boton) {
136                 Toast.makeText(getApplicationContext(), "Premches 'Aceptar'", 1).show();
137             }
138         });
139         venta.setNegativeButton("Cancelar", new DialogInterface.OnClickListener() {
140             public void onClick(DialogInterface dialog, int boton) {
141                 Toast.makeText(getApplicationContext(), "Premeches 'Cancelar'", 1).show();
142             }
143         });
144         return venta.create();
145
146
147
148     case DIALOGO_ENTRADA_TEXTO:
149         // Primeiro preparamos o interior da ventá de diálogo inflando o seu
150         // fichero XML
151         String infService = Context.LAYOUT_INFLATER_SERVICE;
152         LayoutInflater li = (LayoutInflater) getApplicationContext().getSystemService(infService);
153         // Inflamos o compoñente composto definido no XML
154         View inflador = li.inflate(R.layout.dialogo_entrada_texto, null);
155         // Buscamos os compoñentes dentro do Diálogo
156         final TextView etNome = (TextView) inflador.findViewById(R.id.et_nome);
157         final TextView etContrasinal = (TextView) inflador.findViewById(R.id.et_contrasinal);
158
159         venta = new AlertDialog.Builder(this);
160         venta.setTitle("Indica usuario e contrasinal");
161         // Asignamos o contido dentro do diálogo (o que inflamos antes)
162         venta.setView(inflador);
163         // Non se pode incluír unha mensaxe dentro deste tipo de diálogo!!!
164         venta.setPositiveButton("Aceptar", new DialogInterface.OnClickListener() {
165             public void onClick(DialogInterface dialog, int boton) {
166                 Toast.makeText(getApplicationContext(), "Escribiches nome: " + etNome.getText().toString() + "'. Contrasinal: '" +
etContrasinal.getText().toString() + "' e premeches 'Aceptar'",
167                     1).show();
168             }
169         });
170         venta.setNegativeButton("Cancelar", new DialogInterface.OnClickListener() {
171             public void onClick(DialogInterface dialog, int boton) {
172                 Toast.makeText(getApplicationContext(), "Premeches en 'Cancelar'", 1).show();
173             }
174         });
175         return venta.create();
176
177     }
178     return null;
179 }
180
181
182
183 public void onBotonClick(View view) {
184
185     switch (view.getId()) {
186     case R.id.btn_dialogo:
187         showDialog(DIALOGO_MENSAXE);
188         break;
189
190     case R.id.btn_diag_tres_botons:
191         showDialog(DIALOGO_TRES_BOTONS);
192
193         break;
194
195     case R.id.btn_diag_list_selecc:
196         showDialog(DIALOGO_LISTA);
197
198         break;

```

```

199
200     case R.id.btn_diag_radio_button:
201         showDialog(DIALOGO_RADIO_BUTTON);
202
203         break;
204
205     case R.id.btn_diag_checkbox:
206         showDialog(DIALOGO_CHECK_BOX);
207
208         break;
209
210     case R.id.btn_diag_entrada_texto:
211         showDialog(DIALOGO_ENTRADA_TEXTO);
212
213         break;
214
215     default:
216         break;
217 }
218
219 }
220
221
222 @Override
223 public boolean onCreateOptionsMenu(Menu menu) {
224     // Inflate the menu; this adds items to the action bar if it is present.
225     getMenuInflater().inflate(R.menu.u3_10_dialogos, menu);
226     return true;
227 }
228
229 }

```

- **Liñas 20-25:** Creamos constantes enteiras para cada tipo de diálogo que queremos crear.
- Métodos máis comúns para todos os tipos de diálogos:
 - **setTitle():** establece o título da ventá de diálogo.
 - **setMessage():** pon a mensaxe na área de contido da ventá de diálogo.
 - **setIcon():** establece a propiedade Icon cunha das imaxes predefinidas.
- No caso dos cadros de diálogo que teñan botóns:
 - Indicamos para cada tipo de botón o Texto que debe ver o usuario
 - Indicamos a acción a realizar se se preme. Neste caso chámase ao Listener de DialogInterface asociado ao evento Click.
- No método **onBotonClick(View view)** achamos que botón foi o que se pulsou e en función diso chamamos ao método **showDialog()** pasándolle a constante enteira asociada ao diálogo que queremos crear.
- No método **onCreateDialog()** temos asociado un diálogo para cada unha das constantes enteiras. O método recibe o enteiro e crear o diálogo asociado a ese número enteiro.
- Para cada novo diálogo crease un obxecto **venta** co construtor **AlertDialog.Builder()**

Ventás de diálogo personalizadas

- O último diálogo (**Liñas 148-179**) ten asociado un Layout XML. Ficheiro **/res/layout/dialogo_entrada_texto.xml** (xa posto anteriormente).
- Para visualizalo, no canto de utilizar **setMessage()** usamos o método **setView()** (líña 162 do código da activity) para amosar na área de contido do diálogo o XML inflado;
- Ese ficheiro XML hai que instancialo nas súas correspondentes Vistas.
- Ese proceso coñécese co nome de **Inflar** e é necesario facelo para poder visualizar o recurso xml.
- Podemos ver como é dito proceso no código:

```

1     case DIALOGO_ENTRADA_TEXTO:
2         // Primeiro preparamos o interior da ventá de diálogo inflando o seu
3         // ficheiro XML
4         String infService = Context.LAYOUT_INFLATER_SERVICE;
5         LayoutInflater li = (LayoutInflater) getApplicationContext().getSystemService(infService);
6         // Inflamos o compoñente composto definido no XML
7         View inflador = li.inflate(R.layout.dialogo_entrada_texto, null);
8         .....
9         venta.setView(inflador);

```

Neste caso estamos a facer un 'inflate' explícito, pero aínda que non vos dades conta, tamén se fai un 'inflate' cando chamamos o método setContentView dentro do onCreate da activity. Lembrar que o que pasamos como dato nesa chamada é o layout que vai visualizar a activity. Isto se coñece como 'inflate' implícito e o fai o S.O. automaticamente.

- Para nos (aínda que non sexa exactamente iso) o 'inflado' serve para pasar dun recurso de texto xml a un obxecto (View) onde se atopan todos os elementos gráficos que están definidos no arquivo xml. É importante sinalar que cando queiramos acceder a un compoñente do diálogo (por exemplo, no xogo das preguntas, cando engadimos unha pregunta nova aparece un diálogo deste tipo. Ao dar de alta a pregunta leva consigo acceder as caixas de texto) o temos que facer a través do view que obtemos do 'inflado'. Se isto o facemos dende dentro dunha clase anónima, teremos que definir o View como final. No exemplo, o que se define como final é o campo de texto (líñas 156 e 157), pero poderíamos definir como final o View da forma:

```

1      case DIALOGO_ENTRADA_TEXTO:
2          // Primeiro preparamos o interior da ventá de diálogo inflando o seu
3          // ficheiro XML
4          String infService = Context.LAYOUT_INFLATER_SERVICE;
5          LayoutInflater li = (LayoutInflater) getSystemService(infService);
6          // Inflamos o compoñente composto definido no XML
7          final View inflador = li.inflate(R.layout.dialogo_entrada_texto, null);
8
9          venta = new AlertDialog.Builder(this);
10         venta.setTitle("Indica usuario e contrasinal");
11         // Asignamos o contido dentro do diálogo (o que inflamos antes)
12         venta.setView(inflador);
13         // Non se pode incluír unha mensaxe dentro deste tipo de diálogo!!!
14         venta.setPositiveButton("Aceptar", new DialogInterface.OnClickListener() {
15             public void onClick(DialogInterface dialog, int boton) {
16                 // Buscamos os compoñentes dentro do Diálogo
17                 TextView etNome = (TextView) inflador.findViewById(R.id.et_nome);
18                 TextView etContrasinal = (TextView) inflador.findViewById(R.id.et_contrasinal);
19
20                 Toast.makeText(getApplicationContext(), "Escribiches nome: " + etNome.getText().toString() + ".
21                 Contraseñal: " + etContrasinal.getText().toString() + " e premeches 'Aceptar'",
22                     1).show();
23             }
24         });

```

- No inflate o segundo parámetro:

View inflador = li.inflate(R.layout.dialogo_entrada_texto, null);

normalmente ponse null, pero podemos poñer un view que queiramos que sexa 'pai' do view que se xera a partir dos recursos xml.

DialogFragment

A partir da API 13 (Android 3.2) a forma de 'construír' os diálogos modificouse.

Se se usa a forma anteriormente explicada funcionará pero marcará no Compilador a liña como **deprecated**.

```

public void onBotonClick(View view) {
    switch (view.getId()) {
        case R.id.btn_dialogo:
            showDialog(DIALOGO_MENSAXE);
            break;
    }
}

```

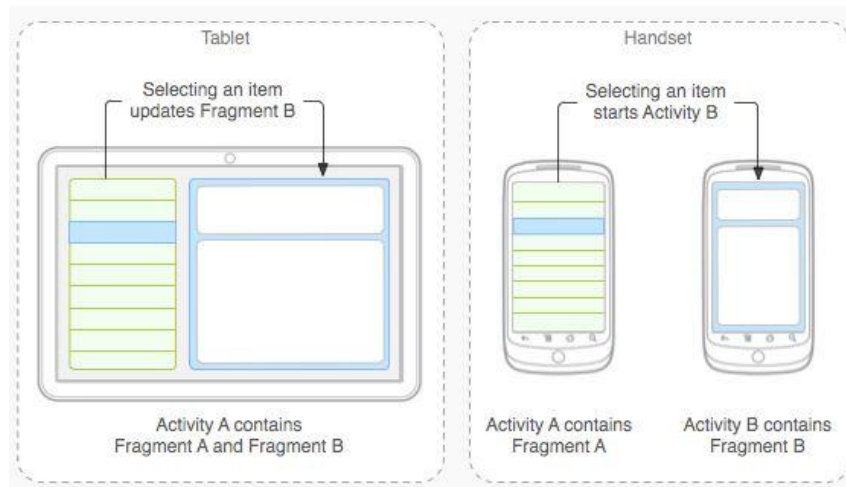
Imos ver neste punto como poderíamos facer para construír os diálogos doutra forma.

Para explicalo teremos que falar antes dos DialogFragment e por extensión dos Fragment.

Información adicional:

- DialogFragment (<http://developer.android.com/reference/android/app/DialogFragment.html#BasicDialog>)
- Fragment (<http://developer.android.com/guide/components/fragments.html>)
- <http://www.sgoliver.net/blog/fragments-en-android/> (En español).

Un fragment representa un 'trozo' da interface dun usuario. A idea dos fragment xurdiu polas dificultades que tiñan as aplicacións a adaptarse a tamaños grandes de pantalla (como as tablet's).



Imaxe obtida de [1] (<http://developer.android.com/guide/components/fragments.html>)

Imaxínade que tedes unha aplicación que amosa unha lista de usuarios e que ó premer sobre un deles aparecieran os seus datos completos.

Isto feito nun móbil cunha pantalla de 4 polgadas levaría consigo a creación de dúas actividades, xa que na mesma non cabe toda a información (necesitaríamos usar scroll, bastante incómodo).

Esta mesma aplicación levada a unha tablet de 10 polgadas non tería problema en visualizar todo na mesma pantalla.

Os fragment vannon permitir dividir os elementos gráficos en 'anacos' que imos poder usar de forma independente nas actividades. Así, no caso anterior podemos crear dous fragment, un coa lista e outro co detalle de cada usuario.

Agora dependendo do tamaño da pantalla pode facer que na pantalla pequena se amose un só fragment (a lista) é que o premer sobre un elemento da mesma apareza o outro fragment (o detalle).

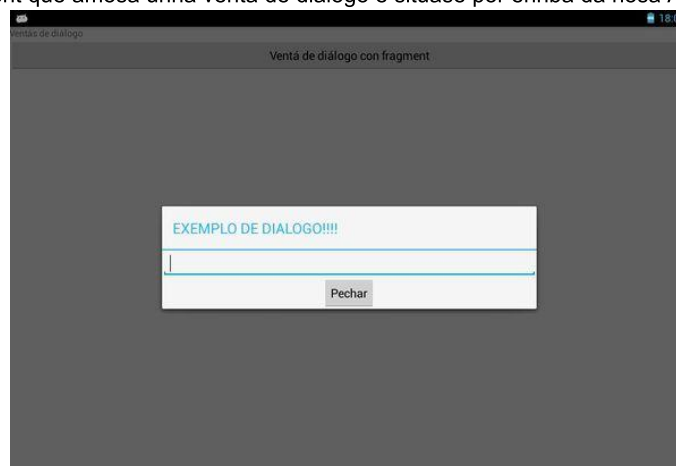
Se o tamaño é grande (tablet) pode facer que aparezan os dous fragment na mesma activity.

O código que xestiona a pulsación dun elemento da lista e amosar os datos estará definido nun só sitio, o que modificamos é o sitio onde se visualiza o fragment en función do tamaño da pantalla.

Tendo claro o concepto de Fragment isto lévanos ao seguinte concepto: **DialogFragment**.

Caso práctico

- Un DialogFragment é un Fragment que amosa unha ventá de diálogo e sitúase por enriba da nosa Activity.



Os DialogFragment poden ser utilizados en versións anteriores á API 13, pero nese caso teremos que utilizar unha biblioteca de compatibilidade (<http://developer.android.com/tools/support-library/index.html>) que nos proporciona Eclipse:



Polo tanto se temos un Min SDK no android manifesto menor á API 13 teremos que importar a clase DialogFragment de dita librería:

```
1 import android.support.v4.app.DialogFragment;
```

Un DialogFragment vai ter un layout asociado que vai compoñer o seu contido.

Definimos polo tanto o contido do noso diálogo.

O XML do DialogFragment

Código da clase fragment_u3_15_layout_dialogo

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:tools="http://schemas.android.com/tools"
4   android:layout_width="match_parent"
5   android:layout_height="match_parent" >
6
7   <EditText
8       android:id="@+id/editTexto"
9       android:layout_height="wrap_content"
10      android:layout_width="match_parent"
11      android:inputType="text"
12      android:gravity="fill_horizontal"
13   />
14   <Button
15       android:id="@+id/buttonPegarDialogo"
16       android:layout_width="wrap_content"
17       android:layout_height="wrap_content"
18       android:layout_below="@+id/editTexto"
19       android:layout_centerInParent="true"
20       android:text="Pegar" />
21
22 </RelativeLayout>
```

Agora deberemos crear unha clase que vai representar o DialogFragment e que vai cargar o layout anterior. Esta clase será a que instanciemos dende a nosa Activity.

O código Java do DialogFragment

Código da clase UD3_15_DialogoFragmento

Obxectivo: Clase que representa ó DialogFragment

```
1 import android.os.Bundle;
2 import android.support.v4.app.DialogFragment;
3 import android.view.LayoutInflater;
4 import android.view.View;
5 import android.view.View.OnClickListener;
6 import android.view.ViewGroup;
7 import android.widget.Button;
8 import android.widget.EditText;
9
10 public class UD3_15_DialogoFragmento extends DialogFragment{
11
12     public String valorTexto;
13
14     @Override
15     public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
16
17         final View rootView = inflater.inflate(R.layout.fragment_u3_15_layout_dialogo, container, false);
18         getDialog().setTitle(getTag()); // O Tag se envía dende a activiy có método show.
19
20         Button btn = (Button) rootView.findViewById(R.id.buttonPegarDialogo);
21         btn.setOnClickListener(new OnClickListener() {
```

```

22
23 24      @Override
25          public void onClick(View v) {
26              // TODO Auto-generated method stub
27              EditText edit = (EditText)rootView.findViewById(R.id.editTexto);
28              valorTexto = edit.getText().toString();
29              ((U3_15_Dialogos)UD3_15_DialogoFragmento.this.getActivity()).pecharDialogo();
30              dismiss();
31          }
32      });
33      // Do something else
34      return rootView;
35  }
36 }

```

Analicemos o código.

- Liña 2: Estamos a utilizar a librería de compatibilidade con versións anteriores de Android.
- Liña 10: A nosa clase deriva de DialogFragment.
- Liña 15: Ó derivar de DialogFragment sobreescribimos o método onCreateView que ten que devolver o View que conforma o diálogo.

No noso caso deberemos 'inflar' o layout deseñado anteriormente.

O obxecto da clase LayoutInflater permite pasar dun deseño gráfico (o layout) e un obxecto manexable por programación. Isto o fai chamando ó método inflate e o garda no obxecto rootView.

- Liña 21: Có paso anterior, podemos acceder ós compoñentes gráficos que se atopan no layout por programación. No noso caso accedemos ó botón pechar e xestionamos o evento de click.
- Liñas 25-31: O que facemos nestas liñas é acceder ó contido da caixa de texto do diálogo, o gardamos nunha propiedade da clase.
 - Liña 28: Esta liña vos dará un erro ata que usedes o código que ven a continuación. Esta liña chama a un método definido na activity que lanzou o DialogFragment.
 - Liña 29: Pechamos o diálogo.

O XML da Activity

Definimos o layout da nosa activity:

Código da clase activity_u3_15_dialogos.xml

Obxectivo: Layout da nosa activity.

```

1  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2      android:layout_width="fill_parent"
3      android:layout_height="fill_parent"
4      android:orientation="vertical" >
5
6      <TextView
7          android:layout_width="match_parent"
8          android:layout_height="wrap_content"
9          android:text="Ventás de diálogo" />
10
11
12      <Button
13          android:id="@+id/btn_dialogo"
14          android:layout_width="match_parent"
15          android:layout_height="wrap_content"
16          android:onClick="onBotonClick"
17          android:text="Ventá de diálogo con fragment" >
18      </Button>
19
20  </LinearLayout>

```

O código Java da Activity

Agora implantaremos a activity que vai facer uso do DialogFragment.

Aquí teremos dúas opcións:

- Derivar a clase de Activity. Deberemos facer uso do método `getFragmentManager()` (<http://developer.android.com/reference/android/app/FragmentManager.html#getFragmentManager%28%29>).

- Derivar a clase de `FragmentActivity`. Esta clase é unha subclase de `Activity` e está posta por compatibilidade coas versións anteriores (<http://developer.android.com/tools/support-library/index.html>).

No caso de utilizar esta clase deberemos facer uso do método `getSupportFragmentManager` (<http://developer.android.com/reference/android/support/v4/app/FragmentManager.html#getSupportFragmentManager%28%29>). **No noso exemplo imos facer uso deste método.**

Código da clase `U3_15_Dialogos`

Obxectivo: Amosar un diálogo baseado en `Fragment`.

```

1 import android.os.Bundle;
2 import android.support.v4.app.FragmentActivity;
3 import android.support.v4.app.FragmentManager;
4 import android.view.View;
5 import android.widget.Toast;
6
7 public class U3_15_Dialogos extends FragmentActivity {
8
9     private UD3_15_DialogoFragmento dialogoFragmento;
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_u3_15_dialogos);
15         dialogoFragmento = new UD3_15_DialogoFragmento();
16     }
17
18     public void pecharDialogo(){
19         Toast.makeText(this, dialogoFragmento.valorTexto, Toast.LENGTH_LONG).show();
20     }
21
22     public void onBotonClick(View view) {
23         FragmentManager fm = getSupportFragmentManager();
24
25         switch (view.getId()) {
26             case R.id.btn_dialogo:
27                 dialogoFragmento.show(fm, "EXEMPLO DE DIALOGO!!!");
28                 break;
29         }
30     }
31 }
32 }

```

Analicemos o código.

- Liñas 2,3,7: Por compatibilidade con versións anteriores de Android escollemos a opción de utilizar un `FragmentActivity`.
- Liña 9,15: Definimos e instanciamos o diálogo.
- Liñas 18-20: Este é o método que chama a clase do Diálogo cando prememos o botón pechar.
- Liña 23: Necesitamos un `FragmentManager` para amosar o diálogo.
- Liña 27: Amosamos o diálogo.

Unha variación

Esta é unha primeira aproximación os `DialogFragment`. Comentar que en vez de sobrescribir o método `onCreateView` do `DialogFragment` podemos sobrescribir o método `onCreateDialog` e devolver un diálogo construído por nos (coma un `DatePickerDialog`, `AlertDialog.builder`,....) Por exemplo:

Código da clase `UD3_15_DialogoFragmento`

Obxectivo: Modificamos o código para amosar outra forma de crear un diálogo. **Debemos comentar o código do método `onCreateView`.**

```

1 import android.app.AlertDialog;
2 import android.app.Dialog;
3 import android.content.DialogInterface;
4 import android.content.DialogInterface.OnClickListener;
5 import android.os.Bundle;
6 import android.support.v4.app.DialogFragment;
7 import android.widget.Toast;
8

```

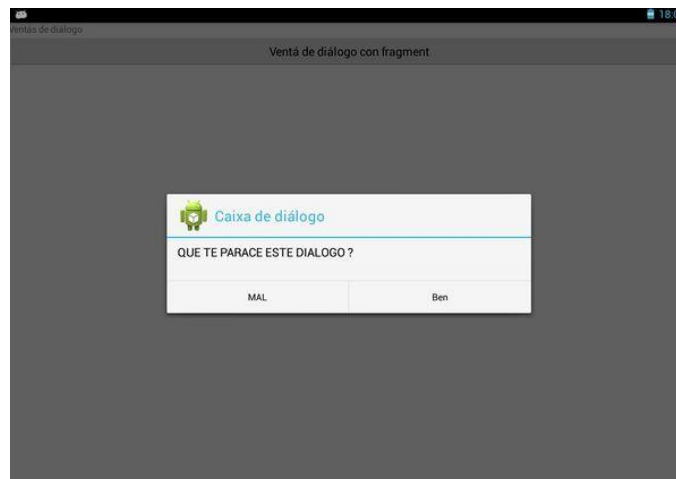


```

9 public class UD3_15_DialogoFragmento extends DialogFragment{
10
11
12     @Override
13     public Dialog onCreateDialog(Bundle savedInstanceState) {
14         AlertDialog.Builder builder = new AlertDialog.Builder(getActivity())
15             .setTitle("Caixa de diálogo").setIcon(R.drawable.ic_launcher)
16             .setMessage("QUE TE PARACE ESTE DIALOGO ?")
17             .setPositiveButton("Ben", new OnClickListener() {
18                 @Override
19                 public void onClick(DialogInterface dialog, int which) {
20                     Toast.makeText(getActivity(), "PULSADA OPCION BOA", Toast.LENGTH_LONG).show();
21                 }
22             }).setNegativeButton("MAL", new OnClickListener() {
23                 @Override
24                 public void onClick(DialogInterface dialog, int which) {
25                     //
26                     Toast.makeText(getActivity(), "PULSADA OPCION MALA", Toast.LENGTH_LONG).show();
27                 }
28             });
29         return builder.create();
30     }
31
32 }
33
34
35 }

```

Nota: Fixarse que estamos a importar a interface `OnClickListener` do `DialogInterface` (líña 4).



-- Ángel D. Fernández González e Carlos Carrión Álvarez -- (2015).

Obtenido de «https://manuais.iessanclemente.net/index.php?title=Ventás_de_Diálogos&oldid=68513»

Esta página se editó por última vez el 5 nov 2020 a las 10:24.

El contenido está disponible bajo la licencia [Creative Commons: CC-BY-NC-SA](#), a menos que se indique lo contrario.