

Aplicaciones Telemáticas

Tema 2.3. Control de la interfaz de usuario (UI) en Android

J. E. López Patiño, F. J. Martínez Zaldívar

ETSIT-UPV

1 Introducción

2 Controles de texto y botón

- Implementando interfaz `View.OnClickListener`
- Otros controles

3 Lanzando actividades

- Actividad lanzando otra actividad

4 Menús de aplicación

Índice

1 Introducción

Introducción

Control del interfaz:

- Controles básicos:
 - TextView
 - Button
 - ToggleButton
 - CheckBox
 - RadioButton
- Arranque de actividades
- Menús de aplicación
- Controles de listas
 - Adaptadores
 - ListView
 - GridView
 - Spinner

Referencia: <http://www.sgoliver.net/blog/>

Índice

2 Controles de texto y botón

- Implementando interfaz `View.OnClickListener`
- Otros controles

Índice

2 Controles de texto y botón

- Implementando interfaz `View.OnClickListener`
- Otros controles

Registro de función de *callback*

- ¿Qué significa `button.setOnClickListener(objeto)`?
`setOnClickListener()`: método público de la clase `View`

`java.lang.Object` \Rightarrow `android.view.View` \Rightarrow `android.widget.TextView` \Rightarrow `android.widget.Button`

```
void setOnClickListener(View.OnClickListener l)
```

Register a callback to be invoked when this view is clicked.

El argumento de `setOnClickListener` debe implementar la interfaz `View.OnClickListener`; en este caso: `objeto`

Help...

```
import android.view.View;
... implements View.OnClickListener {
```

The screenshot shows the Android Developers website in a Chromium browser. The page title is "View.OnClickListener | Android Developers - Chromium". The URL is "developer.android.com/reference/android/view/View.OnClickListener.html". The page has a navigation bar with "Developers", "Design", "Develop", and "Distribute" tabs. Below the navigation bar are links for "Training", "API Guides", "Reference", "Tools", "Google Services", and "Samples". The main content area is titled "View.OnClickListener" and includes a "public static interface" declaration. It also lists "Known Indirect Subclasses" such as "CharacterPickerDialog", "KeyboardView", and "QuickContactBadge". The "Class Overview" section states: "Interface definition for a callback to be invoked when a view is clicked." The "Summary" section is empty. The "Public Methods" section lists the "onClick (View v)" method, which is "Called when a view has been clicked." The "Parameters" section lists "v" as "The view that was clicked." The page also includes a "Menu" section with various Android API classes and a footer with copyright information.

Algunas posibilidades

- Repositorio:

<https://github.com/AATT-ETSIT/U2T3-Ej1-OnClickListener.git>

- Referencias a etiquetas:

- v1.0: argumento de `.setOnClickListener(·)` es un objeto de clase que implementa interfaz `View.OnClickListener`
- v1.1: tanto el objeto anterior como la clase a la que pertenece son anónimos
- v2.0: ahora es `MainActivity` quien implementa la interfaz `View.OnClickListener`
- v3.0: alternativa indicando código a ejecutar en fichero XML
- v4.0: utilización de `vista.getId()` para obtener referencia `R.id.identificador...`

Índice

2 Controles de texto y botón

- Implementando interfaz `View.OnClickListener`
- Otros controles

Button, CheckBox, RadioButton, ToggleButton, EditText...

- Acceso a información y eventos de estos objetos de la clase View
- Ejemplo, repositorio:
<https://github.com/AATT-ETSIT/U2T3-Ej2-TextoYBotones.git>

Índice

3 Lanzando actividades

- Actividad lanzando otra actividad

Índice

3 Lanzando actividades

- Actividad lanzando otra actividad

Actividad \Rightarrow Actividad: esquema general

- Actividad lanzadora:

MainActivity.java

```
...  
  
public class MainActivity extends ...  
  
...  
  
Intent intento = new Intent( MainActivity.this, ActividadLanzada.class );  
  
Bundle b = new Bundle();  
b.putString("NOMBRE_DEL_STRING", "Un string" );  
b.putInt("IDENTIFICADOR_DEL_INT", 7 );  
...  
intento.putExtras( b );  
  
startActivity( intento );  
...
```

- Actividad lanzada

ActividadLanzada.java

```
...  
  
public class ActividadLanzada extends ...  
...  
Bundle bundle = getIntent().getExtras();  
String unString = bundle.getString("NOMBRE_DEL_STRING");  
int x = bundle.getInt("IDENTIFICADOR_DEL_INT");  
...
```

Consideraciones en AndroidManifest.xml

- Es necesario declarar la actividad en AndroidManifest.xml (puede hacerlo el IDE automáticamente)

AndroidManifest.xml

```
...  
<activity android:name=".ActividadLanzada" />  
...
```

- Si deseamos boton UP, entonces tenemos que indicar hacia dónde queremos volver:

AndroidManifest.xml

```
...  
<activity android:name=".ActividadLanzada"  
    android:parentActivityName=".MainActivity" >  
    <!-- The meta-data tag is required if you support API level 15 and lower -->  
    <meta-data  
        android:name="android.support.PARENT_ACTIVITY"  
        android:value=".MainActivity" />  
</activity>  
...
```

Ejemplos con actividades

- URL: <https://github.com/AATT-ETSIT/U2T3-Ej3-A2A.git>
- v1.0: actividad lanzando otra actividad enviando parámetros en un bundle.
- v1.1. Indicación en `AndroidManifest.xml` de actividad padre: Boton UP. Obsérvese diferencia con boton BACK (creación de nueva actividad/recuperación sin destrucción)
- v1.2. Similar con tres actividades y distintas actividades padres (`AndroidManifest.xml`)
- v1.3. Similar, pero haciendo que UP se comporte como BACK.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case android.R.id.home: // UP pulsado
            onBackPressed(); // Acción BACK
            return true;
    }
    return super.onOptionsItemSelected(item);
}
```


Actividad llamando a actividad y esperando resultados

- Actividad lanzadora:

```
...
int CODIGO.PETICION = 5;
Intent intento = new Intent( MainActivity.this, ActividadLanzada.class );

Bundle b = new Bundle();
b.putString("NOMBRE_DEL_STRING", "Un string" );
b.putInt("IDENTIFICADOR_DEL_INT", 7 );
...
intento.putExtras( b );

startActivityForResult( intento, CODIGO.PETICION );
}

@Override
protected void onActivityResult( int codigoPeticion, int codigoResultado, Intent intento ) {
    if ( codigoPeticion == CODIGO.PETICION ) {
        if ( codigoResultado == Activity.RESULT_OK ) {
            int valor_por_defecto = 0;
            String string_por_defecto = "";
            int valor = intento.getIntExtra("RESULTADO.1", valor_por_defecto);
            String cadena = intento.getStringExtra("CADENA", string_por_defecto );
        } else if ( codigoResultado == Activity.RESULT_CANCELED ) {
            ...
        }
    }
}
```

Consdieraciones (cont.)

- Actividad lanzada:

```
...

Bundle bundle = getIntent().getExtras();

String nombreStr = bundle.getString("NOMBRE_DEL_STRING");
int x = bundle.getInt("IDENTIFICADOR_DEL_INT");

...

}

// Código para retornar
{
    ...
    Intent intentoRetorno = new Intent();
    // Alternativa al empleo de bundle
    intentoRetorno.putExtra( "ANYOS", anyos );
    setResult( Activity.RESULT_OK, intentoRetorno );
    finish();
}
```

Ejemplos con actividades, esperando resultados

- URL: <https://github.com/AATT-ETSIT/U2T3-Ej4-A2A-results>
- v1.0: actividad llamando a actividad y esperando resultados
- v1.1: adición de botón UP (sin efecto, salvo vuelta atrás)
- v1.2: con efecto equivalente a validación.
- v1.3: con cancelación explícita
- v1.4: con distintos códigos (distintas actividades llamadas)

Índice

4 Menús de aplicación

Consideraciones

- Sugerencia de plantilla: *Basic Activity* (no *Empty Activity*)
- URL: <https://github.com/AATT-ETSIT/U2T3-Ej5-Menus.git>
- En la plantilla del proyecto aparecen varios *layout* incluyendo uno al otro
- Aparece también el fichero `menu_main.xml` en directorio `menu`
- Etiqueta v1.0: situación inicial con plantilla
- Etiqueta v1.1: adición de algunas variaciones de menú.
- Probar con plantillas *“Navigation Drawer Activity”* y *“Bottom Navigation Activity”*