

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
 Ese apoyo que siempre quiso tener...

1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
 HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
 Gestor de contenidos (Alfresco)
 Aplicaciones híbridas

Tareas programadas (Quartz)
 Gestor documental (Alfresco)
 Inversión de control (Spring)

Control de autenticación y
 acceso (Spring Security)
 UDDI
 Web Services
 Rest Services
 Social SSO
 SSO (Cas)

JPA-Hibernate, MyBatis
 Motor de búsqueda empresarial (Solr)
 ETL (Talend)

Dirección de Proyectos Informáticos.
 Metodologías ágiles
 Patrones de diseño
 TDD

BPM (jBPM o Bonita)
 Generación de informes (JasperReport)
 ESB (Open ESB)



Estás en: Inicio » Tutoriales » Construcción de un control personalizado en Android



DESARROLLADO POR:
Carlos García Pérez

Técnico especialista en informática de empresa (CEU).
Ingeniero Técnico en Informática de Sistemas (UPM)
Creador de MobileTest, Haaala!, Girillo, toi18n.
Charla sobre desarrollo de aplicaciones en Android.
Puedes encontrarme en Autentia: Ofrecemos servicios de soporte a desarrollo, factoría y formación

El e-mail gratuito que estaba esperando.
Almacenamiento de correo electrónico ilimitado y archivos adjuntos hasta 50MB
Regístrese ahora

Fecha de publicación del tutorial: 2011-05-03

2.018

Share |

Regístrate para votar

Construcción de un control personalizado en Android

Introducción

En muchas ocasiones es muy interesante crear nuestros propios controles personalizados para poder reutilizarlos rápida y cómodamente en el futuro. En este tutorial vamos a ver un ejemplo de como crear y usar un control personalizado sencillo.

Manos a la obra! construcción del control personalizado

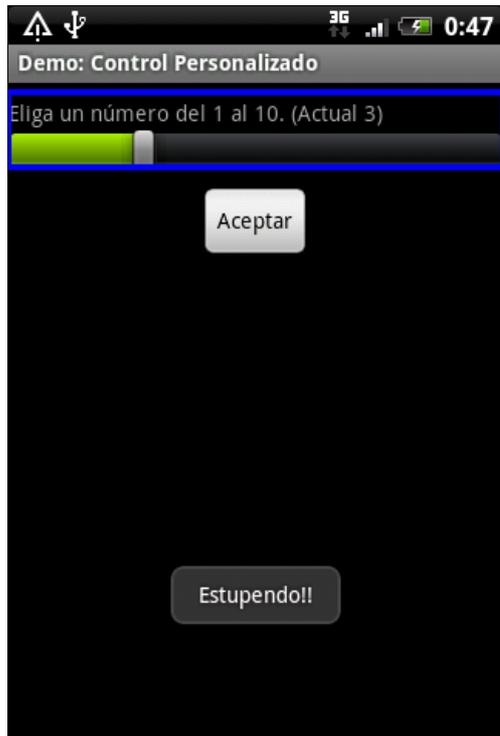
A continuación construiremos un control que nos permita cómodamente elegir mediante una barra de progreso un número entre un determinado intervalo válido, además nos permita personalizar:

- El título
- El valor máximo.
- El valor por defecto seleccionado.

Si deseas el código fuente, puedes [descargarlo aquí](#) . (Está autocomentado, no creo que tengas problemas si tienes una base de programación en Android.)

Captura de pantalla de la aplicación a construir:

El control personalizado que construiremos está resaltado con un borde azul.



Catálogo de servicios Autentia

Últimas Noticias

- VII Autentia Cycling Day
- Autentia patrocina la charla sobre Java SE 7 en Madrid
- Alfresco Day 2011
- XVII Charla Autentia - Grails - Vídeos y Material
- iii 15 millones de descargas de tutoriales !!!

Histórico de NOTICIAS

Últimos Tutoriales

- Spring MVC: acceder a las propiedades de un fichero desde una JSP con Expression Language (EL)
- Framework Scala liftweb
- Trabajando con JAXB y Eclipse
- Configurar Spring Security 3.1 para autenticarse contra un Active Directory
- Migración a ICEfaces 2.0

Últimos Tutoriales del Autor

- Desarrollo de aplicaciones mixtas (web/nativa) en Android
- toi18n, Traduce tus aplicaciones de forma rápida Online
- Haaala! para Android: Facebook y Email con Voz
- Girillo TTS: Una aplicación Android para evitar el uso del móvil durante la conducción
- AppWidget Android: Ejemplo usando BroadcastReceiver y Localización

Síguenos a través de:

Interface gráfico de la aplicación (la captura de pantalla)

Observe como definimos declarativamente la pantalla con el control personalizado es.carlosgarcia.widget.SeekBarCustom que construiremos posteriormente:

```
viewplain print ?
01. <?xml version="1.0" encoding="UTF-8"?>
02. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
03.             android:layout_width="fill_parent" android:layout_height="fill_parent" android:layo
04.
05.             <es.carlosgarcia.widget.SeekBarCustom
06.                 android:id="@+id/barId" android:text="@string/barTitle"
07.                 android:max="10" android:progress="5"
08.                 android:layout_width="fill_parent" android:layout_height="wrap_content" android:layo
09.
10.             <Button android:id="@+id/button1" android:text="@string/accept"
11.                  android:layout_height="wrap_content" android:layout_width="wrap_content"
12.                  android:layout_gravity="center_horizontal" android:layout_marginTop="10dip"/>
13. </LinearLayout>
14.
```

es.carlosgarcia.MainActivity

Único Activity de la aplicación.

```
viewplain print ?
01. package es.carlosgarcia;
02.
03. import android.app.Activity;
04. import android.os.Bundle;
05. import android.view.View;
06. import android.view.View.OnClickListener;
07. import android.widget.Button;
08. import android.widget.Toast;
09. import es.carlosgarcia.widget.SeekBarCustom;
10.
11. /**
12.  * Control personalizado que representa una barra de progreso con un título con:
13.  * @param min: Valor mínimo
14.  * @param progress: Valor actual
15.  * @param text: Título mostrado
16.  * @author Carlos García. http://www.carlos-garcia.es
17.  */
18. public class MainActivity extends Activity implements OnClickListener {
19.     private Button btn;
20.     private SeekBarCustom seek;
21.
22.     @Override
23.     public void onCreate(Bundle savedInstanceState) {
24.         super.onCreate(savedInstanceState);
25.
26.         this setContentView(R.layout.main);
27.         this.btn = (Button) this.findViewById(R.id.button1);
28.         this.seek = (SeekBarCustom) this.findViewById(R.id.barId);
29.         this.btn.setOnClickListener(this);
30.     }
31.
32.     /**
33.      * Cuando hacemos clic en el botón invocamos un método personalizado y específico del co
34.      */
35.     @Override
36.     public void onClick(View v) {
37.         if (seek.isValueSelected()){
38.             Toast.makeText(this, "Estupendo!!", Toast.LENGTH_LONG).show();
39.         } else {
40.             Toast.makeText(this, "Seleccione un valor, por favor.", Toast.LENGTH_LONG).show(
41.         }
42.     }
43. }
```

es.carlosgarcia.widget.SeekBarCustom

Clase que define el control personalizado que deseamos construir (lo resaltado en azul en la captura de pantalla anterior).

Observe como:

- Construimos el control **combinando (o componiendo)** varios controles ya definidos (LinearLayout, SeekBar y TextView)
- **Leemos los atributos** que hemos definido en el XML del GUI. (max, progress, text).

```
viewplain print ?
01. package es.carlosgarcia.widget;
02.
03. import android.content.Context;
04. import android.util.AttributeSet;
05. import android.widget.LinearLayout;
06. import android.widget.SeekBar;
07. import android.widget.SeekBar.OnSeekBarChangeListener;
08. import android.widget.TextView;
09.
```

Últimas ofertas de empleo

- 2011-07-06  Otras Sin catalogar - LUGO.
- 2011-06-20  Comercial - Ventas - SEVILLA.
- 2011-05-24  Contabilidad - Especialista Contable - BARCELONA.
- 2011-05-14  Comercial - Ventas - TARRAGONA.
- 2011-04-13  Comercial - Ventas - VALENCIA.

```

10.  /**
11.   * Control personalizado que representa una barra de progreso con un título con:
12.   * @param min:      Valor mínimo
13.   * @param progress: Valor actual
14.   * @param text:     Título mostrado
15.   * @author Carlos García. http://www.carlos-garcia.es
16.   */
17.  public class SeekBarCustom extends LinearLayout implements OnSeekBarChangeListener {
18.      private static final String NS      = "http://schemas.android.com/apk/res/android";
19.      private static final String PROGRESS = "progress";
20.      private static final String MAX     = "max";
21.      private static final String TEXT    = "text";
22.
23.      private int      resCode;
24.      private SeekBar  bar;
25.      private TextView caption;
26.
27.      public SeekBarCustom(Context context, AttributeSet attrs) {
28.          super(context, attrs);
29.          this.initUI(attrs);
30.      }
31.
32.      /**
33.       * Inicializamos el control leyendo los atributos personalizados max, progress y text
34.       * @param attrs
35.       */
36.      private void initUI(AttributeSet attrs){
37.          this.caption = new TextView(this.getContext());
38.          this.bar      = new SeekBar(this.getContext());
39.          this.resCode = attrs.getAttributeResourceValue(NS, TEXT, 0);
40.
41.          this.setOrientation(LinearLayout.VERTICAL);
42.          this.addView(caption);
43.          this.addView(bar);
44.
45.          int progress = attrs.getAttributeUnsignedIntValue(NS, PROGRESS, 1);
46.          caption.setText(getResources().getString(resCode, progress));
47.
48.          bar.setMax(attrs.getAttributeUnsignedIntValue(NS, MAX, 100) + 1);
49.          bar.setProgress(progress);
50.
51.
52.          bar.setOnSeekBarChangeListener(this);
53.      }
54.
55.      /**
56.       * Permite al usuario establecer el valor, validamos que sea un valor correcto.
57.       */
58.      public void setProgress(int p){
59.          if ((p < 0) || (p > bar.getMax())){
60.              throw new IllegalArgumentException();
61.          }
62.          bar.setProgress(p);
63.      }
64.
65.      public int getProgress(){
66.          return bar.getProgress();
67.      }
68.
69.      /**
70.       * Indica si el usuario ha seleccionado algún valor
71.       */
72.      public boolean isValueSelected(){
73.          return (bar.getProgress() > 0);
74.      }
75.
76.      /**
77.       * Cambiamos el título del control cuando el usuario mueve el SeekBar.
78.       */
79.      @Override
80.      public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
81.          caption.setText(getResources().getString(resCode, progress));
82.      }
83.
84.      @Override
85.      public void onStartTrackingTouch(SeekBar seekBar) {}
86.
87.      @Override
88.      public void onStopTrackingTouch(SeekBar seekBar) {}
89.  }
90.
91.

```

Archivo de recursos de mensajes /res/string.xml

Nota: Si desea traducirlo automáticamente a otros idiomas, puede hacerlo a mano o usar la aplicación [toi18n](#).

```

view plain print ?
01.  <?xml version="1.0" encoding="UTF-8"?>
02.  <resources>
03.      <string name="app_name">Demo: Control Personalizado</string>
04.      <string name="url">http://www.carlos-garcia.es</string>
05.      <string name="barTitle">Eliga un número del 1 al 10. (Actual %1$d)</string>
06.      <string name="accept">Aceptar</string>

```

Conclusiones

Como veis no es difícil crear estos controles y reutilizarlos en un futuro o incluso crearos vuestras propias librerías de componentes..

Espero que os haya sido útil, un saludo.
Carlos García, <http://www.carlos-garcia.es>.

Anímate y coméntanos lo que pienses sobre este **TUTORIAL**:

Puedes opinar o comentar cualquier sugerencia que quieras comunicarnos sobre este tutorial; con tu ayuda, podemos ofrecerte un mejor servicio.

Enviar comentario

(Sólo para usuarios registrados)

>> **Registrate** y accede a esta y otras ventajas <<

COMENTARIOS



Esta obra está licenciada bajo licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5

Copyright 2003-2011 © All Rights Reserved | [Texto legal y condiciones de uso](#) | [Banners](#) | [Powered by Autentia](#) | [Contacto](#)

W3C XHTML 1.0

W3C CSS

XML RSS

XML ATOM