

Somos su empresa de Soporte a Desarrollo Informático

Ese apoyo que siempre quiso tener

- Desarrollo de componentes y proyectos a medida.
- Auditoría de código y recomendaciones de mejora.
- Arranque de proyectos basados en nuevas tecnologías.
- Curso de Formación
 - Dirección de Proyectos Informáticos.
 - Gestión eficaz del Tiempo.
 - Arquitecturas de desarrollo Web: Web, J2EE, SOA, WebServices, BPM, etc.
 - Java/ J2EE a todos los niveles: JSPs, Servlets, EJBs, JMS, JNI, etc.
 - Análisis y diseño orientado a objeto.
 - UML y patrones de diseño.
 - Buenas prácticas en el desarrollo de aplicaciones
 - Técnicas avanzadas: Lucene, Hibernate, Spring, JSF, Struts, etc.

*Nuestra mejor referencia son los conocimientos que
compartimos en nuestro web*

www.adictosaltrabajo.com

Decenas de entidades cuentan ya con nosotros

Para más información visítenos en www.autentia.com

Tel. 91 675 33 06 - info@autentia.com



Ponente: Carlos García Pérez



- Programación en dispositivos de recursos limitados.
- Introducción a Android
 - ¿Qué es Android?
 - Arquitectura de Android
 - Aplicaciones en Android
 - Instalación
 - Desinstalación
 - Android Market
- Desarrollo de aplicaciones en Android
 - Herramientas de desarrollo
 - SDK
 - ADT: Pluggin para el IDE Eclipse



- Estructura de las aplicaciones en Android
 - Estructura de directorios.
 - El archivo AndroidManifest.xml
 - Recursos e internacionalización.
 - Interfaz gráfico (GUI)
 - Actividades
 - Comunicación entre actividades/reutilización de actividades externas.
 - BroadcastReceiver.
 - Proveedores de contenido.
 - Servicios.
- Preguntas/Dudas/Comentarios

Programac. en dispositivos con RL



www.autentia.com

- **Recursos limitados:** memoria, almacenamiento, CPU y batería.
- **Esforzarse en desarrollar bien** desde el primer momento.
 - **No se puede agregar más hardware** como un S^a tradicional.
 - Usar herramientas como PMD, CPD para aprender a mejorar.
 - Vector, concatenación de cadenas, StringBuilder sin capacidad inicial, lectura completa de archivos o sin buffer, copy/paste, sincronización deficiente, gestión de excepción deficiente, liberar recursos.
- Hay que hacer las aplicaciones **usables** (además no tenemos 17")
- Hay que tener en mente **todas las situaciones posibles**, hay que **probar mucho**.
 - ¿Y si no hay conectividad de red? > Al menos mostrar un Alert



Introducción a Android

¿Qué es Android?



www.autentia.com

- **SO + Suite extensa de aplicaciones y librerías + VM**
- **Diseñado** para dispositivos con **recursos limitados**, libre y “totalmente abierta” (¿VM?) para todos (Fabricantes, desarrolladores).
 - Libertad de modificar y adaptar el SO (licencia Apache, v2.).
- Creado por la **Open Handset Alliance** (encabezada por **Google**).
- **Limitación baja para los desarrolladores** (tu imaginación)
 - Finalizar llamadas salientes cuando pasan 10 segundos y no son contestadas => ¿Cómo con J2ME?
 - Enviar un SMS con un texto clave y dirección de correo => envíe localización sin que el usuario se enteré si quiera (me roban el dispositivo)
- **Crecimiento exponencial** > Según AdMob en el último año 2% al 38%....

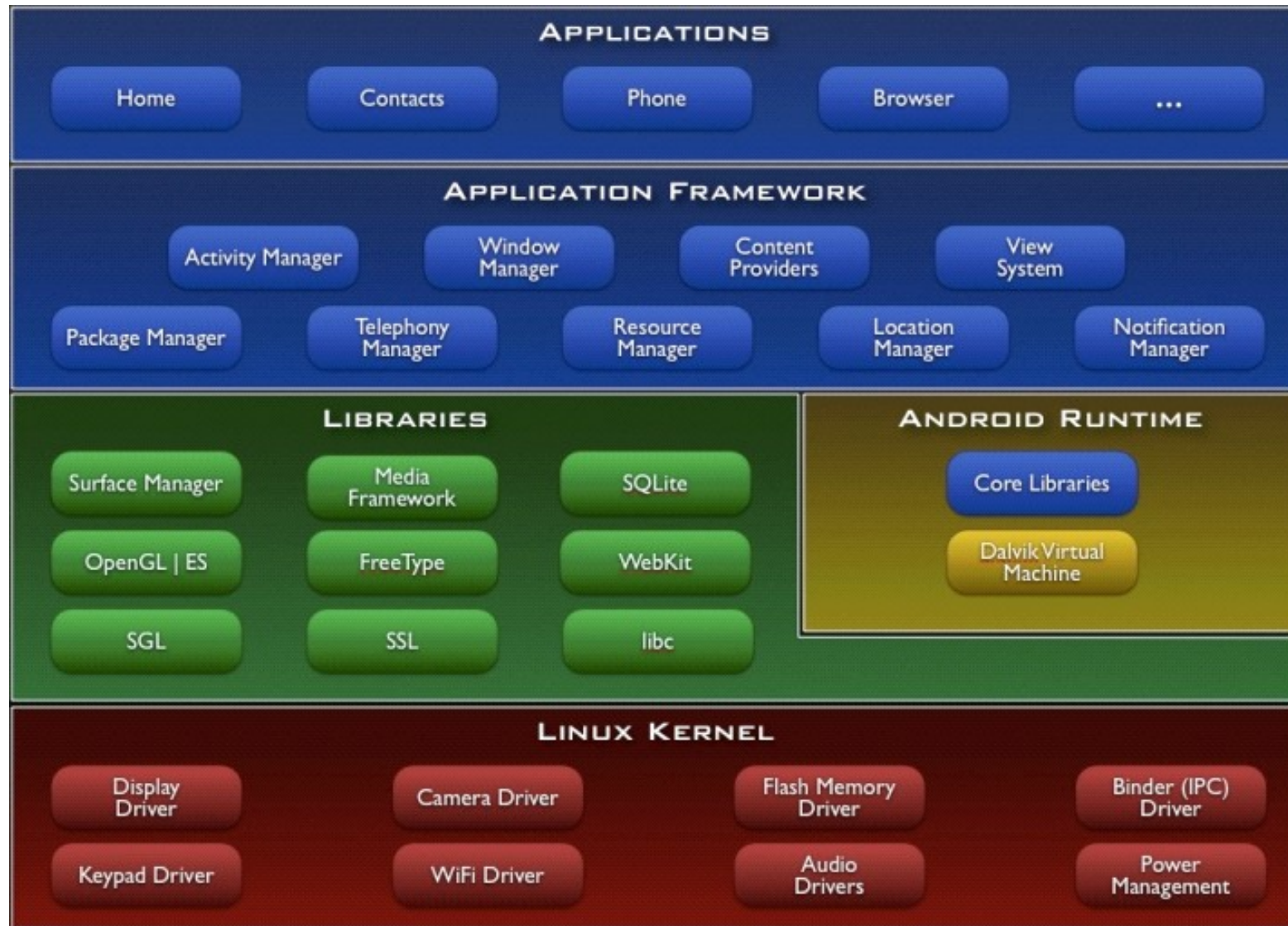
¿Qué es Android? (continuación)



www.autentia.com

- **Nuevas versiones** cada poco tiempo, **en pocos tiempo** han salido las siguientes versiones:
 - 1.1, 1.5, 1.6, 2.0, 2.1, 2.2, 2.0.1
 - Desarrollar con la versión **más baja posible**
- En cada nueva versión:
 - Se corrigen posibles Bugs.
 - Se agregan nuevas aplicaciones de serie para el usuario.
 - Mejora el API para el programador
 - Mejoras de rendimiento y capacidades (por ejemplo, reproducción de nuevos formatos de audio)
 - <http://developer.android.com/sdk/android-1.5-highlights.html>

Arquitectura de Android



Aplicaciones en Android



- **apk** => Aplicación
- **Extenso** API Java
 - Sintaxis 1.5
- **JARs externos** (.class!!)
 - DEX
- Cada apk => Proceso independiente => **VM** independiente => Memoria independiente => Datos **privados** por apk

javax.net
javax.net.ssl
javax.security.auth
javax.security.auth.callback
javax.security.auth.login
javax.security.auth.x500
javax.security.cert
javax.sql
javax.xml
javax.xml.datatype
javax.xml.namespace
javax.xml.parsers
javax.xml.transform
javax.xml.transform.dom
javax.xml.transform.sax
javax.xml.transform.stream
javax.xml.validation

package
javax.xml.parsers
Classes | [Description](#)

Provides classes allowing the pro
[more...](#)

Classes

DocumentBuilder
DocumentBuilderFactory
SAXParser
SAXParserFactory

Exceptions

ParserConfigurationException

Instalación de aplicaciones



- Varias alternativas:
 - Android Market.
 - *%SDK_HOME%/tools/adb install <path_apk>* (USB)
 - A través del **plugin de Eclipse** (USB)
 - **Desde instaladores** disponibles en el Market. (Buscar palabras como apk, installer).
 - Las instalaciones por USB, web, email,.. (desde otros lugares distintos al Market) requieren tener activada la opción “**Origenes desconocidos**”(Ajustes => Aplicaciones)

Desinstalación de aplicaciones



- Varias alternativas:
 - ***%SDK_HOME%/tools/adb uninstall <paquete>***
 - Desde el GUI de Android => Ajustes -> Aplicaciones -> Administración de aplicaciones.
 - Desde desinstaladores disponibles en el Market.

Android Market



www.autentia.com

- **Cualquier desarrollador** puede publicar en el Market, previo pago (25 dolares aprox)
- Incluso aplicaciones **no firmadas por CA.**
- **No hay límite** sobre el número de aplicaciones a instalar.
- **% de la venta** para Google.
- **No se revisan las aplicaciones** que se suben, será el propio público el que las haga famosas y las descarte (con sus votaciones y comentarios).
- **APK malintencionada** => Responsable el autor.
- **¿Te instalarías un apk desde redes p2p, webs?** Yo, no.



Android Market (cont.)



- **Sistema de actualizaciones** de versiones.
- Publicar => Seleccionar una **categoría, un título, descripción y una captura de pantallas**. (En varios idiomas)
- Es todo en **tiempo real, no hay esperas**.
- Las aplicaciones instaladas **no se borran físicamente**, sóloamente se pueden despublicar (deshabilitar).
- El Market usa como número de versión el atributo versión **VersionCode** (los usuarios ven el **VersionName**).

El Market de Android (cont.)



The screenshot shows the 'Application Error Reports' section of the Android Market developer console. It displays data for the application 'com.girillo' (Applications: Multimedia). The data is organized into two columns: 'Freezes' and 'Crashes'. Each column has two rows: 'new' and 'old'. The 'new' row shows '0 new' and '0 reports' with a progress bar, and '0 reports/week'. The 'old' row shows '0 old' and '0 reports' with a progress bar, and '0 reports/week'. The status 'Terminado' is visible at the bottom left of the console window.

Application	Freezes		Crashes	
	new	reports/week	new	reports/week
com.girillo	0 new	0 reports/week	0 new	0 reports/week
Applications: Multimedia	0 reports		0 reports	
	0 old	0 reports/week	0 old	0 reports/week
	0 reports		0 reports	



Desarrollo de aplicaciones en Android

Herramientas de desarrollo - SDK



www.autentia.com

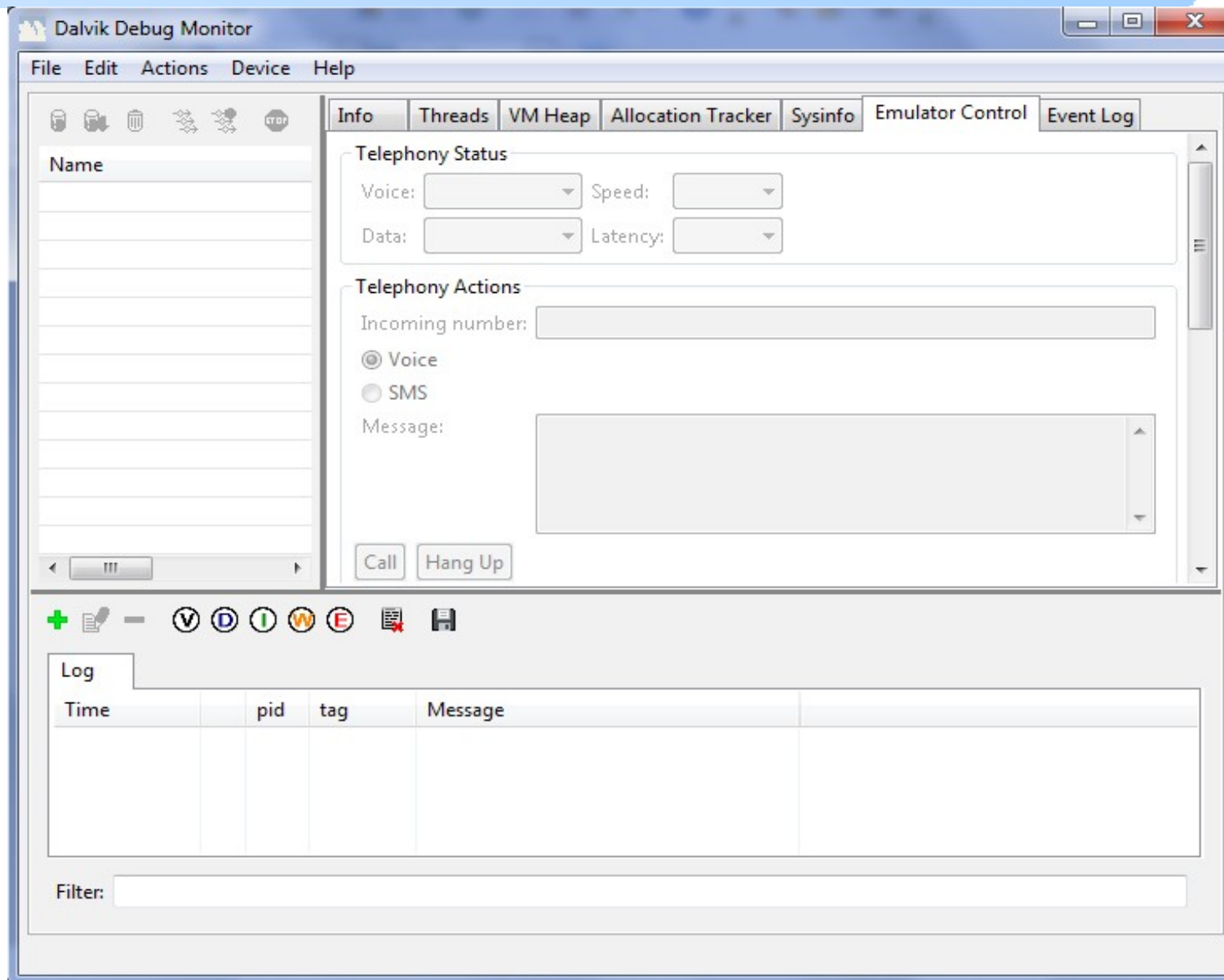
El SDK contiene:

- Documentación.
- API para todas las versiones de Android.
- Ejemplos.
- Driver USB.
[Ajustes>Aplicaciones>Desarrollo>Depuración USB]
- Herramientas para empaquetar aplicaciones, instalar, depurar, configurar entorno emulación
 - **dx**: Convierte .class en .dex
 - **aapt** (Android Asset Packaging Tool): Empaqueta binarios y recursos en un apk
 - **mskcard**: Creación de sdcard, **sqlite3**: Cliente de bases de datos SQLite.
 - `android create project --package com.autentia.HelloWorld --activity HelloActivity --target 2 --path HelloAndroid`

%SDK_PATH%/tools/ddms



www.autentia.com



%SDK_PATH%/tools/adb



- Android Debug Bridge: (Instalar aplicaciones [memoria interna], abrir shell linux, copiar archivos pc/dispositivo, mostrar logs, procesos,...)

- *adb devices*

```
emulator-5554 device
```

```
emulator-5556 device
```

```
- adb -s emulator-5556 install unaAplicacion.apk
```

```
- adb -s emulator-5554 push a.txt /sdcard/a.txt
```

```
- adb -s emulator-5554 pull /sdcard/a.txt a.txt
```

```
- adb -s emulator-5556 forward tcp:6100 tcp:7100
```

```
- adb -s emulator-5556 logcat
```

```
- adb -s emulator-5556 shell ls /system/bin
```

```
- ¿Quieres saber más? >> adb help
```

- Dirección loopback es 10.0.2.2 (No 127.0.0.1)

%SDK_PATH%/tools/adb (cont.)



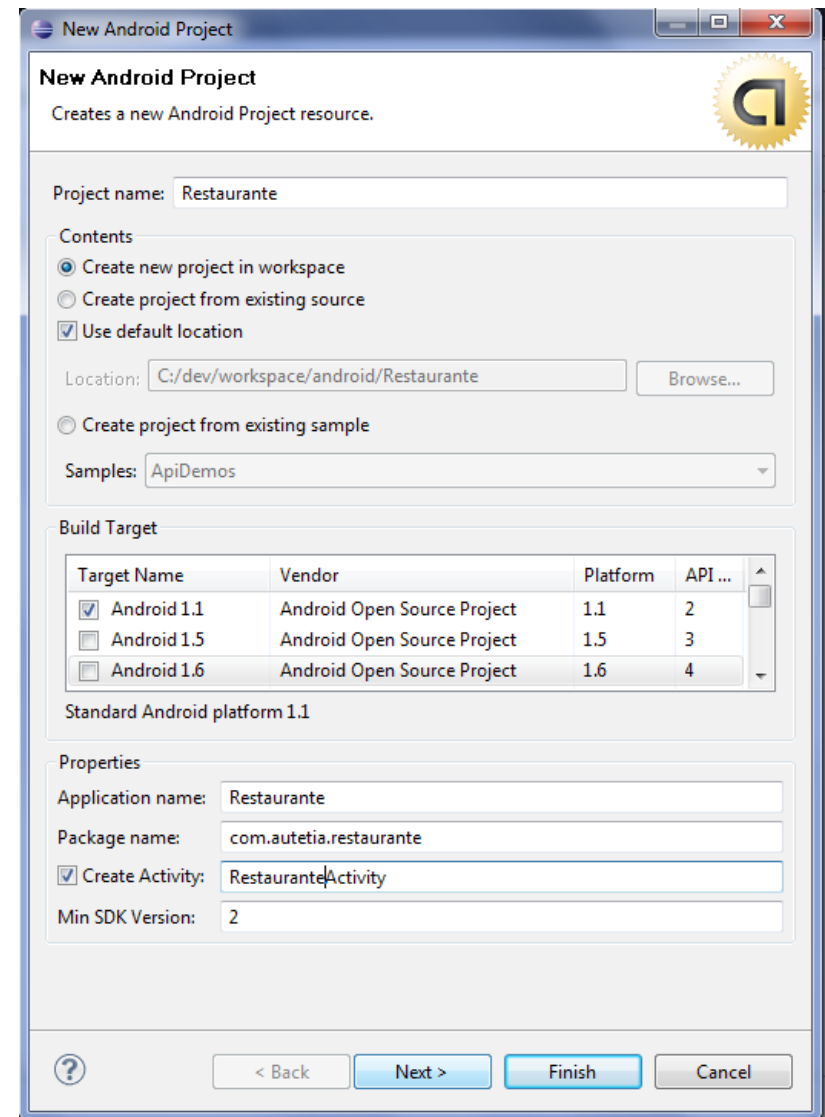
```
Administrator: C:\Windows\system32\cmd.exe - adb shell
$ cd sdcard
cd sdcard
$ ls -l
ls -l
drwxrwxrwx system system 2010-06-16 23:59 DCIM
drwxrwxrwx system system 2009-07-24 17:45 MP3
-rwxrwxrwx system system 1820 2009-06-10 21:49 checksum.txt
drwxrwxrwx system system 2010-01-17 18:24 andnav2
-rwxrwxrwx system system 45808440 2009-07-08 10:27 HTCsync.exe
-rwxrwxrwx system system 1575 2009-06-10 20:52 md5sum.txt
drwxrwxrwx system system 2009-12-26 19:39 rosie_scroll
drwxrwxrwx system system 2009-12-26 19:40 albumthumbs
drwxrwxrwx system system 2010-06-16 23:22 download
drwxrwxrwx system system 2010-02-11 19:23 AgileMessenger
drwxrwxrwx system system 2010-02-17 22:19 maps
drwxrwxrwx system system 2010-05-15 17:12 com.tokasiki.android.voicerecorder
drwxrwxrwx system system 2010-01-11 14:19 espeak-data
drwxrwxrwx system system 2010-03-05 17:13 z7logs
-rwxrwxrwx system system 21471 2010-03-12 19:08 mylog.txt
-rwxrwxrwx system system 4318 2010-04-12 19:16 untitled-[2]
drwxrwxrwx system system 2010-04-25 15:43 phoneintentbundle
drwxrwxrwx system system 2010-04-26 18:32 urbanstew.RehearsalAssistant
drwxrwxrwx system system 2010-06-17 00:04 girillo
```

Herramientas de desarrollo - Eclipse



www.autentia.com

- El plugin de Eclipse (ADT)
- Install => New Software:
 - <https://dl-ssl.google.com/android/eclipse/>



IDE Eclipse (cont)



The screenshot shows the Eclipse IDE interface with the Emulator Control panel open. The panel is divided into several sections:

- Telephony Status:** Includes dropdown menus for Voice (home), Speed (GPRS), Data (home), and Latency (GPRS).
- Telephony Actions:** Includes an input field for Incoming number (699221570), radio buttons for Voice and SMS (SMS is selected), and a text area for Message (Bla Bla Bla Bla). Buttons for Send and Hang Up are present.
- Location Controls:** Includes tabs for Manual, GPX, and KML. Radio buttons for Decimal (selected) and Sexagesimal are present. Input fields for Longitude (-122,084095) and Latitude (37,422006) are shown, along with a Send button.

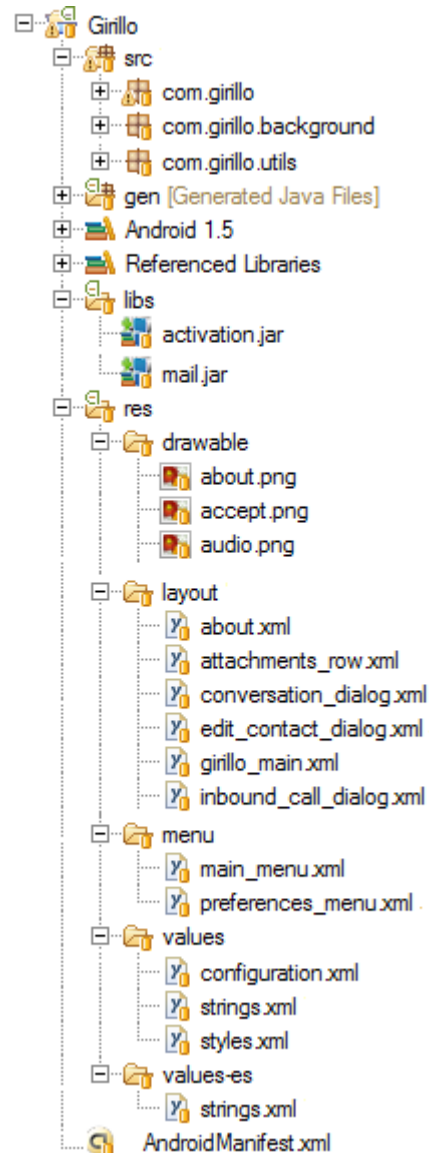
IDE Eclipse (cont)



The screenshot shows the Eclipse IDE interface with the LogCat window open. The LogCat window displays a list of log messages for the 'girillo' application. The messages include system boot logs, service startup logs, and application-specific logs. The messages are color-coded by severity: red for errors, yellow for warnings, green for information, and blue for debug.

Time	pid	tag	Message
06-19 10:23:01.068	W 587	HeadsetObserver	This kernel does not have wired k
06-19 10:23:01.068	I 587	SystemService	Starting AppWidget Service
06-19 10:23:01.168	I 587	WindowManager	Menu key state: 0 safeMode=false
06-19 10:23:01.198	I 587	WindowManager	Config changed: { scale=1.0 imsi=
06-19 10:23:01.238	D 587	PowerManager...	system ready!
06-19 10:23:01.258	W 587	ActivityManager	Unable to start service Intent {
06-19 10:23:01.258	E 587	LockPatternK...	Failed to bind to GLS while check
06-19 10:23:01.418	D 587	dalvikvm	GC freed 3691 objects / 223912 by
06-19 10:23:01.838	W 587	ResourceType	No package identifier when gettir
06-19 10:23:01.929	W 587	ResourceType	No package identifier when gettir
06-19 10:23:01.969	W 587	ResourceType	No package identifier when gettir
06-19 10:23:02.018	D 587	ActivityManager	Start running!
06-19 10:23:02.098	D 560	qemud	fdhandler_accept_event: accepting
06-19 10:23:02.098	D 560	qemud	created client 0xc038 listening c

Estructura de las apps en Android



AndroidManifest.xml



- Describe todos los componentes de la aplicación.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.autentia" android:versionName="2.1" android:versionCode="4">
    <application android:icon="@drawable/icon" android:label="@string/app_name"
        android:debuggable="true">
        <activity android:name=".SplashActivity"
            android:label="@string/app_name"
            android:noHistory="true"
            android:screenOrientation="portrait"
            android:screenOrientation="nosensor"
            android:launchMode="singleInstance">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity android:name=".ContactActivity"/>
    </application>
</manifest>
```

(Continua...)

AndroidManifest.xml (cont.)



```
<receiver android:name="com.autentia.OnBootBroadcastReceiver">
    <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED" />
    </intent-filter>
</receiver>
<service android:name="com.autentia.ListenCallService"/>
</application>

<uses-sdk android:minSdkVersion="3" />

<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.CAMERA"/>
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_CONTACTS"/>
<uses-permission android:name="android.permission.INTERNET"/>

<uses-feature android:name="android.hardware.camera"/>
<uses-feature android:name="android.hardware.microphone"/>
<uses-feature android:name="android.hardware.location.gps" android:required="false"/>
</manifest>
```

Recursos e Internacionalización



www.autentia.com

- Cada idioma/país en su carpeta específica.

– Ejemplos:

- `res/values/strings.xml`
- `res/values-it/strings.xml`
- `res/values-es/strings.xml`
- `res/drawable/magic_icon.png`
- `res/drawable-it/magic_icon.png`

- Desde Java accedemos via:

– `R.resource_type.resource_name`

Ejemplos:

```
this.setContentView(R.layout.about);  
alertDialog.setIcon(R.drawable.delete);
```

Recursos e Internacionalizac. (cont)



The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays the project structure for 'AudioByMail', with the 'strings.xml' file selected under the 'res/values-es' directory. The main editor window shows the XML content of 'strings.xml' with the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="app_name">Aplicación XXXXX</string>
  <string name="about">Acerca de</string>
  <string name="exit">Salir</string>
  <string name="startRecord">Iniciar Grabación</string>
  <string name="stopRecord">Detener Grabación</string>
  <string name="sendRecord">Enviar Grabación</string>
</resources>
```

Interfaz gráfico (GUI)



- Contrucción **programática** o **declarativamente**.
- Ubicados en los paquetes **android.widget** y **android.view**.
- Controles con los que interactua el usuario, entre otros:
 - Lista (Spinner).
 - Casillas de verificación (Checkbox).
 - Imagen (ImageView).
 - Botón (Button, ToggleButton).
 - Cuadro de diálogo (AlertDialog).
 - Menú (Menu).
 - Autocompletado (AutoCompleteTextView).

Interfaz gráfico (GUI) (cont.)



- Podemos **contruir controles personalizados y reutilizarlos**:
 - `<com.autentia.charla.android.MyControl`
- Cada control responde a **todo tipo de eventos**:
 - Teclado, foco, clics, click largos, etc.
- **Se organizan usando Layout**:
 - RelativeLayout, TableLayout, LinearLayout, ScrollLayout, FrameLayout, etc.

Interfaz gráfico (GUI) (cont.)

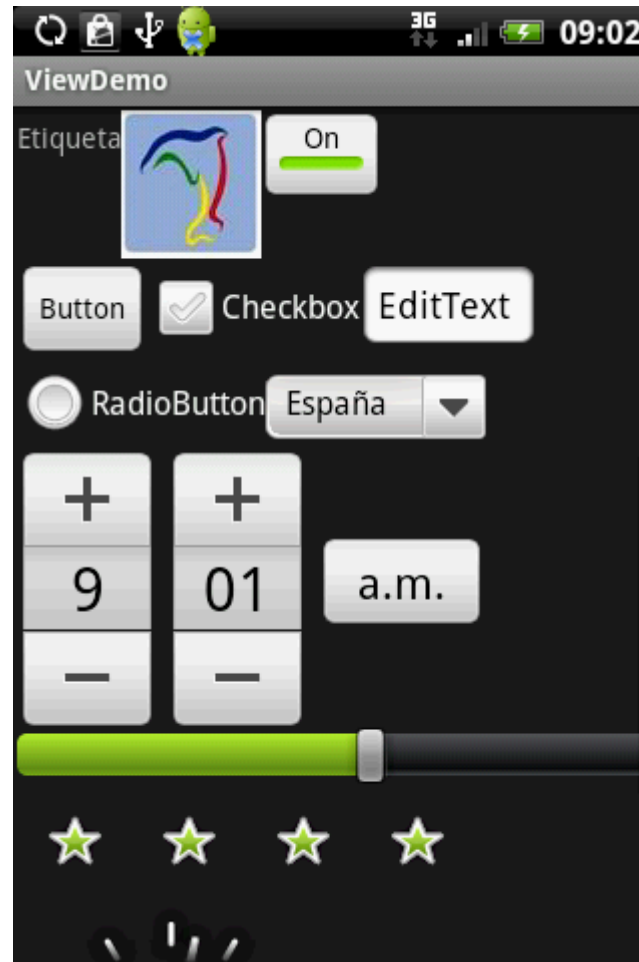


```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="fill_parent" android:layout_height="fill_parent" android:padding="2sp">
    <LinearLayout android:orientation="horizontal" android:layout_width="fill_parent" android:layout_height="wrap_content">
        <TextView android:layout_width="wrap_content" android:layout_height="wrap_content" android:text="Etiqueta"/>
        <ImageView android:src="@drawable/delfin" android:layout_width="wrap_content" android:layout_height="wrap_content"/>
        <ToggleButton android:textOn="On" android:textOff="Off" android:layout_width="wrap_content" android:layout_height="wrap_content"/>
    </LinearLayout>
    <LinearLayout android:orientation="horizontal" android:layout_width="fill_parent" android:layout_height="wrap_content">
        <Button android:text="Button" android:layout_width="wrap_content" android:layout_height="wrap_content"/>
        <CheckBox android:text="Checkbox" android:layout_width="wrap_content" android:layout_height="wrap_content"/>
        <EditText android:text="EditText" android:layout_width="wrap_content" android:layout_height="wrap_content"/>
    </LinearLayout>
    <LinearLayout android:orientation="horizontal" android:layout_width="fill_parent" android:layout_height="wrap_content">
        <RadioButton android:text="RadioButton" android:layout_width="wrap_content" android:layout_height="wrap_content"/>
        <Spinner android:layout_width="wrap_content" android:layout_height="wrap_content" android:prompt="@string/pais"
            android:entries="@array/paises"/>
    </LinearLayout>
    <ScrollView android:layout_width="fill_parent" android:layout_height="wrap_content">
        <LinearLayout android:orientation="vertical" android:layout_width="fill_parent" android:layout_height="fill_parent">
            <TimePicker android:layout_width="wrap_content" android:layout_height="wrap_content"/>
            <SeekBar android:layout_height="wrap_content" android:layout_width="fill_parent" android:max="100"/>
            <RatingBar android:layout_width="wrap_content" android:layout_height="wrap_content" android:numStars="4"/>
            <AnalogClock android:layout_width="wrap_content" android:layout_height="wrap_content"/>
        </LinearLayout>
    </ScrollView>
</LinearLayout>
```

Interfaz gráfico (GUI) (cont.)



www.autentia.com



Actividades (Activity)



- Las **ventanas** con las que interactúa el usuario.
- Cada **actividad es independiente del resto**, están totalmente desacopladas.
- **Intents** construir el flujo de la aplicación: Una actividad puede invocar otras actividades tanto de la misma aplicación como de otra...
 - **Método de comunicación**, representan **una necesidad** [necesito hacer esto con estos datos] (Explícitos e implícitos)
 - **No reinventaremos la rueda**, por ejemplo, si una aplicación desea enviar un email usará la aplicación especializada en el envío de emails, selección de contactos, mostrar mapas etc.
- Actividad principal de la aplicación => MAIN.
- `activity.setContentview()`

Actividades (continuación)



```
public class MainActivity extends android.app.Activity {
    @Override
    public void onCreate(android.os.Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        this setContentView(R.layout.main);

        Button accept = (Button) this.findViewById(R.id.accept);
        accept.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                /* Tratamiento del evento */
            }
        })
    }
}
```

Comunicación entre actividades



- Paso 1) Iniciamos la Actividad2 desde la Actividad1:

```
startActivityForResult(new Intent(this, Actividad2.class), COD_OPERAC);
```

- Paso 2) La Actividad2 hace lo que tenga que hacer y finaliza:

```
Intent intent = new Intent();  
intent.putExtra("FILE_PATH", filePath);  
this.setResult(Activity.RESULT_OK, intent); // RESULT_CANCELED  
this.finish();
```

Comunicac. entre actividades (cont)



- Paso 3) La Actividad1 toma el control y lee el resultado de la siguiente forma:

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == COD_OPERAC){
        if (resultCode == Activity.RESULT_CANCELED) {
            .....
        } else if (resultCode == Activity.RESULT_OK) {
            String filePath = data.getStringExtra("FILE_PATH");
            .....
        }
    }
}
```

Comunicac. entre actividades (cont)



- ¿Y si tengo que transmitir tipos no primitivos?

```
public class Persona implements android.os.Parcelable {
    private String email, name;
    public Persona(android.os.Parcel parcel) {
        this.name = parcel.readString();
        this.email = parcel.readString();
    }
    /* Getters y Setters */
    public void writeToParcel(android.os.Parcel dest, int flags) {
        dest.writeString(this.name);
        dest.writeString(this.email);
    }
    public static final Parcelable.Creator<Persona> CREATOR =
        new Parcelable.Creator<Persona>() {
            public Persona createFromParcel(android.os.Parcel in) { return new Persona(in); }

            public Persona[] newArray(int size) {return new Persona[size];}
        };
}
```

Reutilización de activity externas



Ejemplos (con resolución implícita)

- `startActivity(new Intent(Intent.ACTION_VIEW, Uri.fromFile("/sdcard/autentia.png", "image/png")));`
- `startActivity(new Intent(Intent.ACTION_VIEW, Uri.fromFile("/sdcard/video.mp4", "video/mp4")));`
- `startActivity(new Intent(Intent.ACTION_DIAL, Uri.parse("tel: 91111111")));`
- `startActivity(new Intent(Intent.VIEW_ACTION, Uri.parse("http://www.autentia.com")));`
- `startActivity(new Intent(Intent.ACTION_EDIT, Uri.parse("content://contacts/people/101")));`

Reutilización de activity externas



- ```
Intent intent = new Intent(Intent.ACTION_SEND, "text/plain");
intent.putExtra(Intent.EXTRA_SUBJECT, "asunto del email");
intent.putExtra(Intent.EXTRA_TEXT, "cuerpo del email");
intent.putExtra(Intent.EXTRA_EMAIL, new String[]{"<email1>"});
intent.putExtra(Intent.EXTRA_STREAM, "file://rutaAlArchivo");
startActivity(intent);
```
- **Otros:** Google Maps, Google Street View...
- ¿Nuestra aplicación es muy buena en algo? Podríamos abrirla a aplicaciones terceras de manera que **reaprovecharan su funcionalidad => Componente reutilizable.**
- ¿Qué pasa si hay **varios componentes capaces de procesar una determinada necesidad?** > Android consulta al usuario.
- Por supuesto, también podemos usar **startActivityForResult**

# BroadcastReceiver



- Posibilidad de escuchar **mensajes o eventos** en forma de `Intents` que genera la plataforma Android o las aplicaciones incluso cuando la aplicación no se está en ejecución.
- Subclases de `android.content.BroadcastReceiver`.
- Tu aplicación pueden indicar el deseo de ser notificadas:
  - **Programáticamente:** `Context.registerReceiver()`
  - **Declarativamente:** `<receiver>` `AndroidManifest.xml`
- Ejemplos de eventos generados por Android:
  - `android.intent.action.ACTION_BATTERY_LOW`
  - `android.intent.action.ACTION_HEADSET_PLUG`
  - `android.intent.action.BOOT_COMPLETED`
  - `android.provider.Telephony.SMS_RECEIVED`
  - `android.intent.action.GTALK_CONNECTED`
  - `android.net.wifi.WIFI_STATE_CHANGED`



# BroadcastReceiver (continuación)



- **AndroidManifest.xml:**

```
<receiver android:name="com.autentia.OnBootBroadcastReceiver" android:enabled="true">
 <intent-filter>
 <action android:name="android.intent.action.BOOT_COMPLETED" />
 </intent-filter>
</receiver>
....
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
</manifest>
```

- **Lógica de negocio:**

```
public class OnBootBroadcastReceiver extends android.content.BroadcastReceiver {
 @Override
 public void onReceive(Context context, Intent intent) {
 if ("android.intent.action.BOOT_COMPLETED".equals(intent.getAction())) {
 // Podríamos por ejemplo, iniciar un servicio de monitorización de red
 }
 }
}
```

# BroadcastReceiver (continuación)



[www.autentia.com](http://www.autentia.com)

El proyecto de ejemplo:

BroadcastReceiverDemo

Monitorización del estado de la WIFI

# BroadcastReceiver (continuación)



- Los mensajes broadcast se pueden clasificar en dos tipos:
  - a) **Normales:** (Ejemplo: La pantalla se ha apagado)
    - Enviados via `Context.sendBroadcast`.
    - **Asíncronos**, el orden de recepción es desconocido.
    - No se puede leer su **resultado**.
    - No se puede **abortar** su ejecución.
  - b) **Ordenados:** (Ejemplo: Cancelación de llamadas salientes)
    - Enviados via `Context.sendOrderedBroadcast`.
    - **Síncronos**, el orden de recepción es conocido (si tienen la misma prioridad)
    - Se pueden cancelar, leer y modificar el **resultado**.
      - `abortBroadcast()`, `setResultData`, Modificando el `Budlet` del `Intent`.

# Proveedores de contenido



www.autentia.com

- **¿Compartir** datos entre aplicaciones?
  - `ContentProvider`, `ContentResolver`
  - Altas, bajas, modificaciones y consultas
  - `ContentProvider` listos para usar:
    - **CallLog**: Detalles sobre el historial de llamadas.
    - **Contacts**: Agenda de contactos (tlfes, emails, etc).
    - **MediaStore**: Metadata de archivos multimedia.
    - **Browser**: Datos del Navegador (historial, marcadores)
- **¿Cómo** comparto datos? **Implementan una interfaz.**
  - Métodos para consultar, agregar, modificar y eliminar datos.
- Una **URL distinta por cada tabla** que se exponga.
  - `content://com.autentia.charla/notas`
  - `content://com.autentia.charla/eventos`

# Proveedores de contenido (cont.)



- Ejemplo, **agregar un nuevo contacto:**

```
import android.provider.Contacts.People;
import android.content.*;

....

....
ContentValues values = new ContentValues();
values.put(People.NAME, "Nombre del contacto");
values.put(People.STARRED, 1); // ¿Es un contacto favorito?
Uri uri = getContentResolver().insert(People.CONTENT_URI, values);
....

....
```

# Proveedores de contenido (cont.)



- **Ejemplo, consultar el historial de llamadas:**

```
Cursor callCursor = getContentResolver().query(
 android.provider.CallLog.Calls.CONTENT_URI,
 null, null, null, android.provider.CallLog.Calls.DATE + " DESC");
// Url, Columns, Where, WhereArgument, Order By

startManagingCursor(callCursor);

while (callCursor.moveToNext()) {
 String num = callCursor.getString(callCursor.getColumnIndex(Calls.NUMBER));
 int callDate = callCursor.getInt(callCursor.getColumnIndex(Calls.DATE));
 int callType = callCursor.getInt(callCursor.getColumnIndex(Calls.TYPE));
 int duration = callCursor.getInt(callCursor.getColumnIndex(Calls.DURATION));

}

stopManagingCursor(callCursor);
```



- Componente que permite realizar tareas incluso cuando el **resto de la aplicación no está en ejecución**. (Ej un Player MP3)
- **No tienen** interfaz gráfica de usuario.
- **No es** un Thread, **no es** un proceso independiente de la app.
- Clase que hereda de: `android.app.Service`
- Definido en `AndroidManifest` mediante `<service>`
- Métodos a destacar:
  - `Context.startService(Intent)`: Inicia un servicio.
  - `Context.stopService(Intent)`: Detiene un servicio
  - `StopSelf()`: El servicio se detiene el mismo
- ¿Necesitamos comunicarnos con el de modo síncrono?
  - Listener si estamos dentro de la misma aplicación.
  - Mediante `android.os.Binder`.

# Servicios (continuación)



```
public class DemoService extends Service implements Runnable {
 private ServerSocket socket;
 private Thread thread;

 public void onCreate() {
 super.onCreate();
 this.thread = new Thread(this);
 }

 public void onStart(android.content.Intent intent, int startId) {
 super.onStart(intent, startId);
 if ((thread == null) || (! thread.isAlive())){
 thread.start();
 }
 }

 ...
}
```



# Servicios (continuación)



[www.autentia.com](http://www.autentia.com)

...

```
@Override
public android.os.IBinder onBind(android.content.Intent intent) {
 return null;
}
```

```
@Override
public void onDestroy() {
 // ... ¿ tareas de liberación de recursos ? ...
 super.onDestroy();
}
```

...

# Servicios (continuación)



```
public void run() {
 try {
 this.socket = new ServerSocket(9999, 10);
 while (true) {
 Socket sock = this.socket.accept();
 // Tareas a realizar cuando se conecte un usuario
 }
 } catch (Exception ex) {
 Log.e(DemoService.class.getName(), ex.toString());
 } finally {
 Log.e(DemoService.class.getName(), "Exit Service");
 // ... tareas de liberación de recursos ...
 }
} // End run()
} // End ServiceDemo
```

# Preguntas/Dudas/Comentarios



[www.autentia.com](http://www.autentia.com)



ANDROID

