

Programación Básica en Android

#android2013

09/05/2013

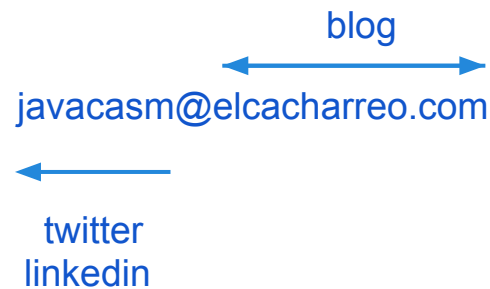


ElCacharreo.com

Programación en Android



José Antonio Vacas Martínez



Programación en Android: Recursos

ANDROID developer.android.com

Muy bueno http://www.sgoliver.net/blog/?page_id=3011

Avanzado <http://www.limecreativelabs.com/curso-gratuito-de-desarrollo-para-android/>

Avanzado(En) <http://www.vogella.com/android.html>

MiriadaX http://miriadax.net/es/web/android_programacion



Programación en Android: Objetivos

- Conocer la plataforma Android
- Crear proyecto de aplicación básica
- Crear "pantallas"



Programación en Android: ¿qué sabes?

¿qué sabes de Android?

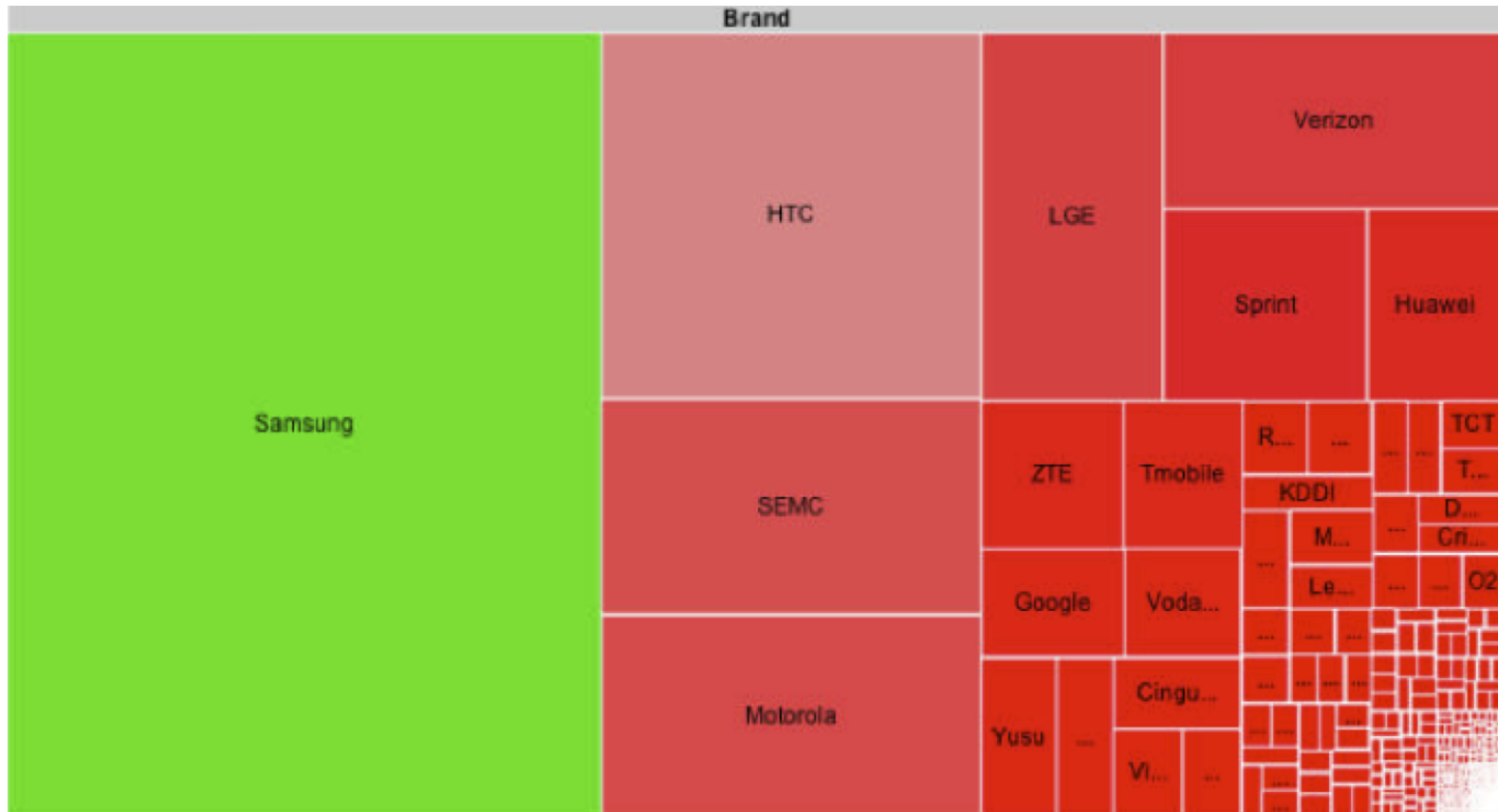
¿qué teléfono tienes?

¿qué usas?

¿para qué lo usas?

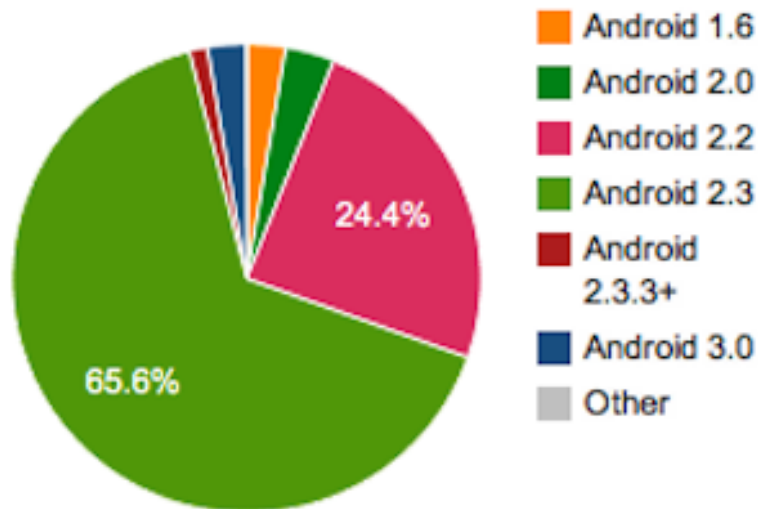


Programación en Android: Fabricantes

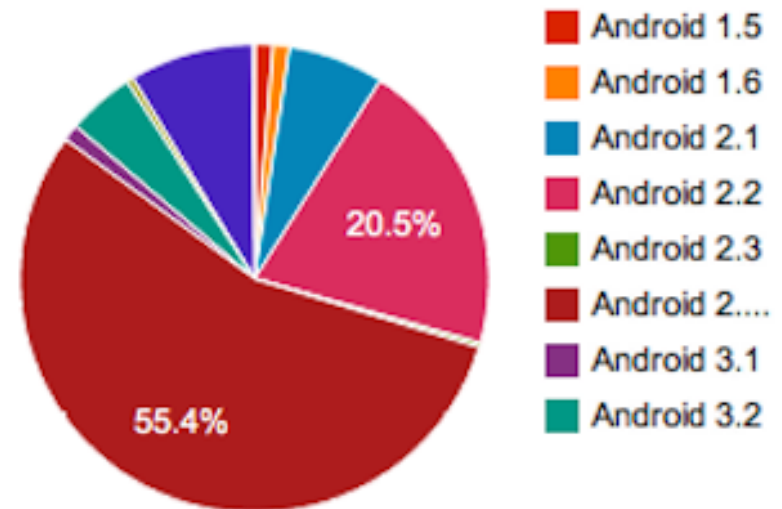


Programación en Android: Versiones

API levels April 2011



API levels April 2012



@opensignalmaps

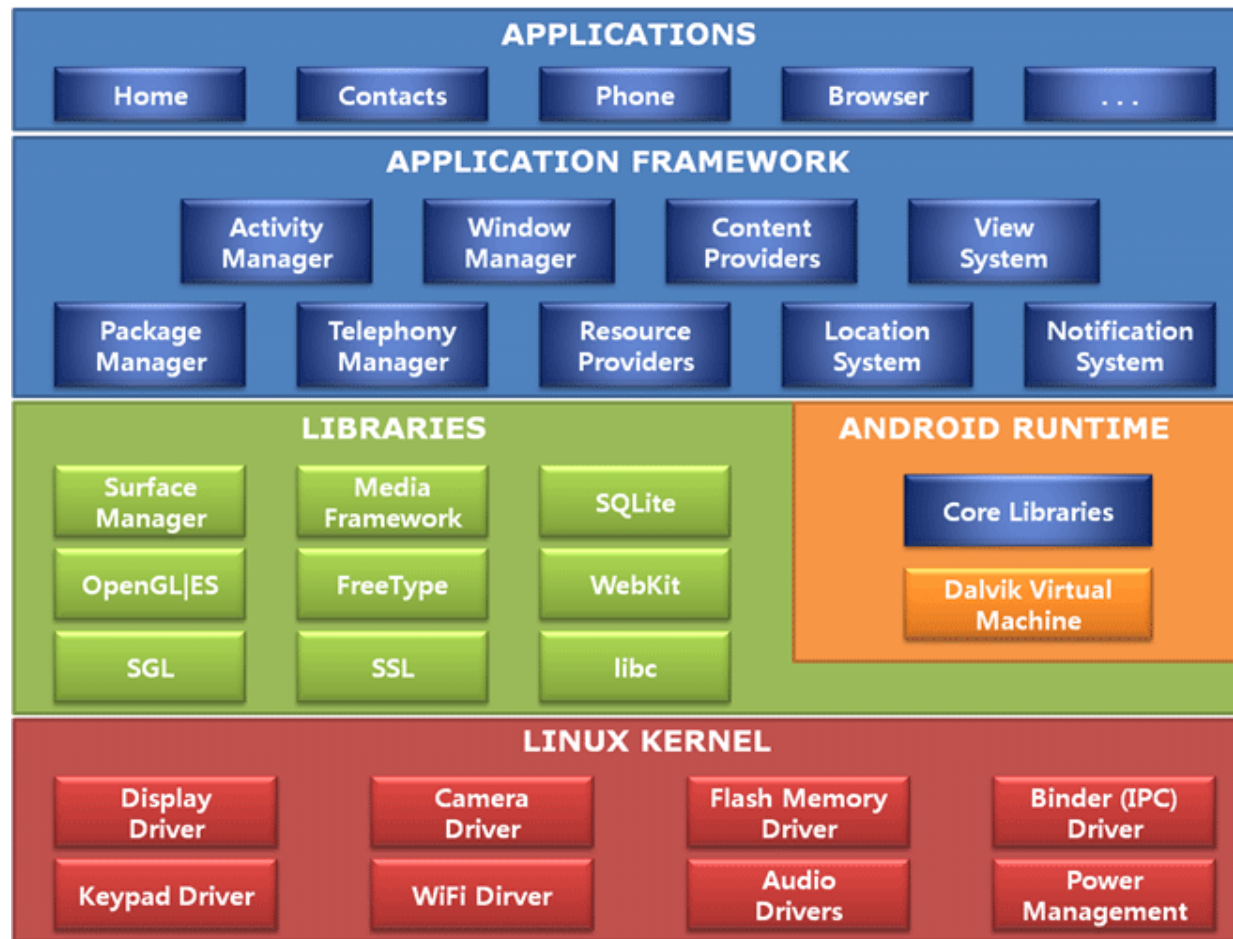


Programación en Android: Versiones

Version	Codename	API	Distribution
1.6	Donut	4	0.2%
2.1	Eclair	7	2.4%
2.2	Froyo	8	9.0%
2.3 - 2.3.2	Gingerbread	9	0.2%
2.3.3 - 2.3.7		10	47.4%
3.1	Honeycomb	12	0.4%
3.2		13	1.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	29.1%
4.1	Jelly Bean	16	9.0%
4.2		17	1.2%



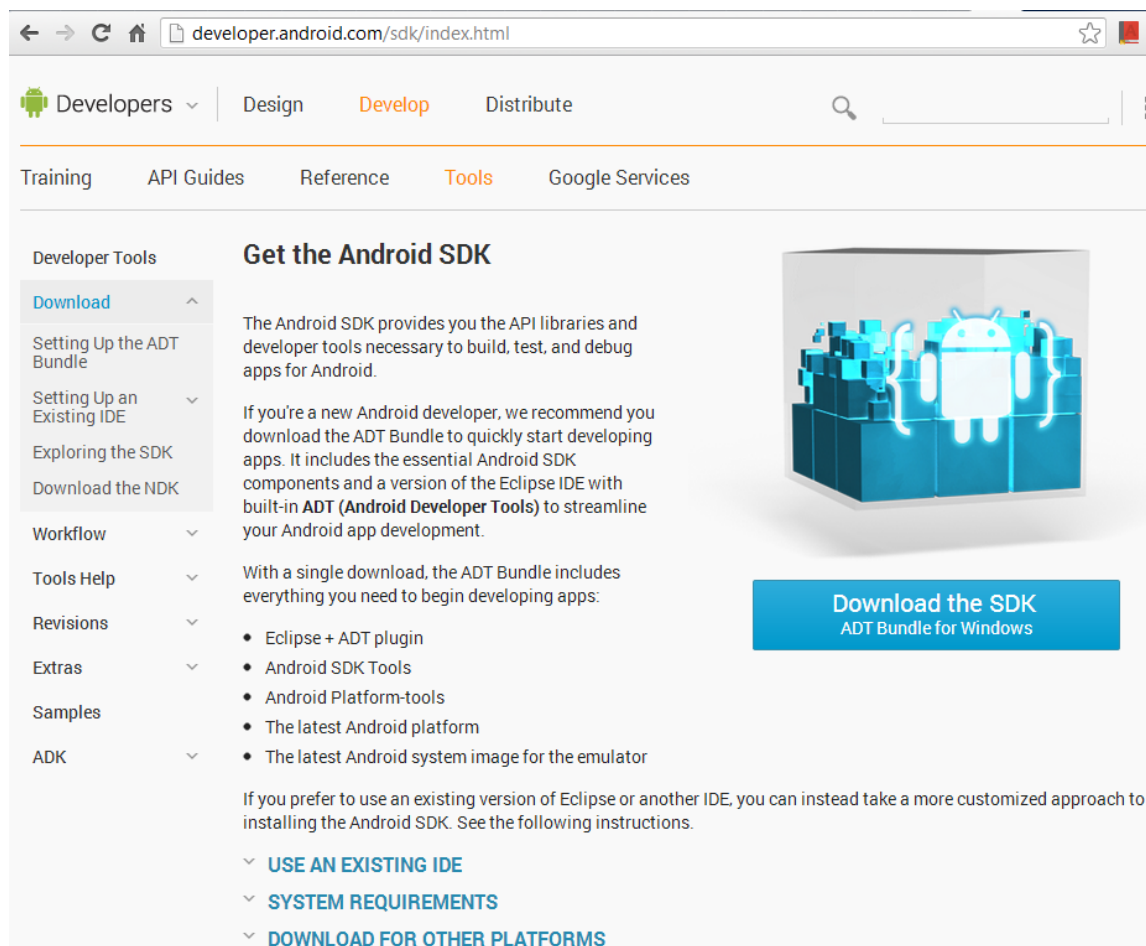
Programación en Android: Arquitectura



[Detalle](#)



Programación en Android: Instalando



The screenshot shows the 'Get the Android SDK' page on the developer.android.com website. The page has a navigation bar with 'Design', 'Develop', and 'Distribute' tabs, and a search bar. Below the navigation bar, there are links for 'Training', 'API Guides', 'Reference', 'Tools', and 'Google Services'. The main content area is titled 'Get the Android SDK' and includes a description of the SDK, a list of components included in the ADT Bundle, and a button to 'Download the SDK ADT Bundle for Windows'. The left sidebar contains a list of links under 'Developer Tools'.

Developer Tools

- Download
- Setting Up the ADT Bundle
- Setting Up an Existing IDE
- Exploring the SDK
- Download the NDK
- Workflow
- Tools Help
- Revisions
- Extras
- Samples
- ADK

Get the Android SDK

The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android.

If you're a new Android developer, we recommend you download the ADT Bundle to quickly start developing apps. It includes the essential Android SDK components and a version of the Eclipse IDE with built-in **ADT (Android Developer Tools)** to streamline your Android app development.

With a single download, the ADT Bundle includes everything you need to begin developing apps:

- Eclipse + ADT plugin
- Android SDK Tools
- Android Platform-tools
- The latest Android platform
- The latest Android system image for the emulator

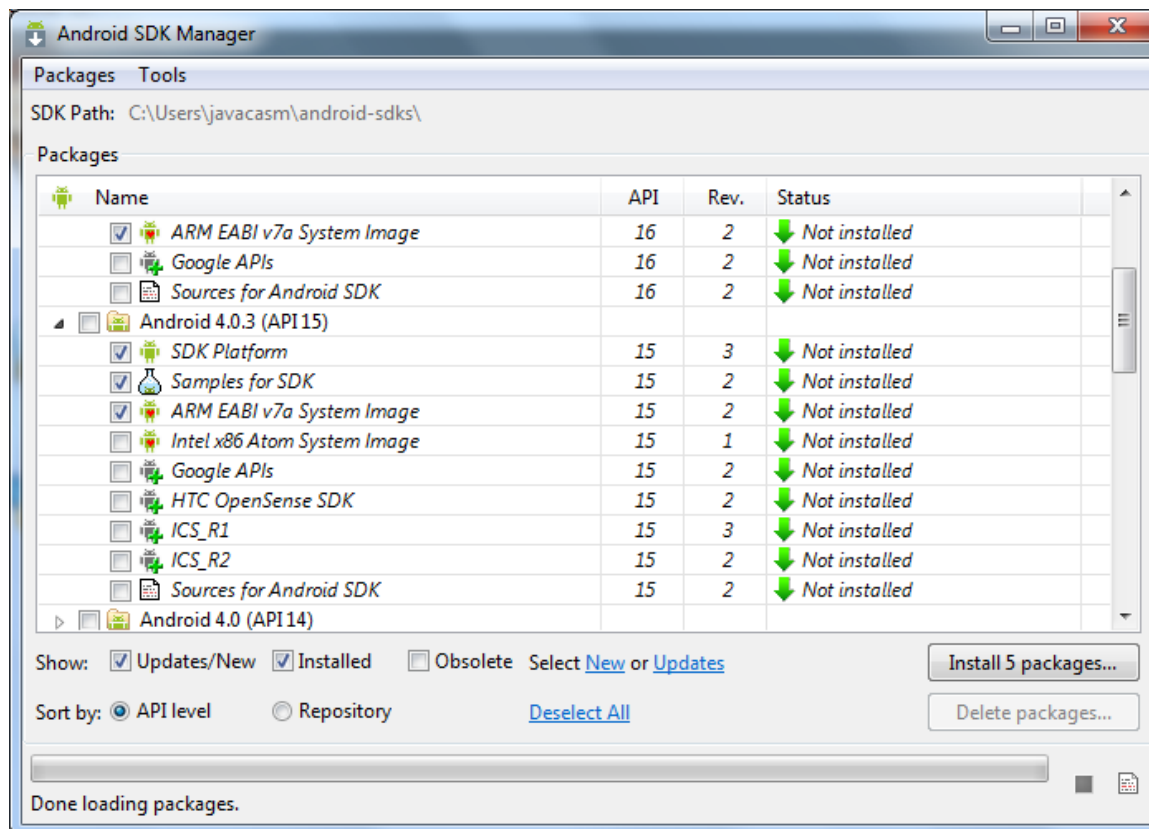
If you prefer to use an existing version of Eclipse or another IDE, you can instead take a more customized approach to installing the Android SDK. See the following instructions.

- [USE AN EXISTING IDE](#)
- [SYSTEM REQUIREMENTS](#)
- [DOWNLOAD FOR OTHER PLATFORMS](#)

[Download the SDK ADT Bundle for Windows](#)



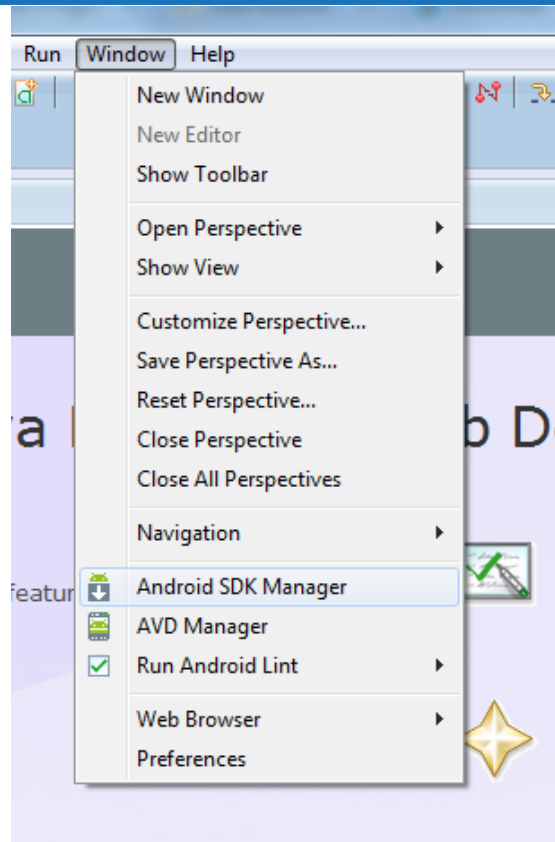
Programación en Android: SDKs



Instalamos la versión 4.0.3 adecuada para usar tablets



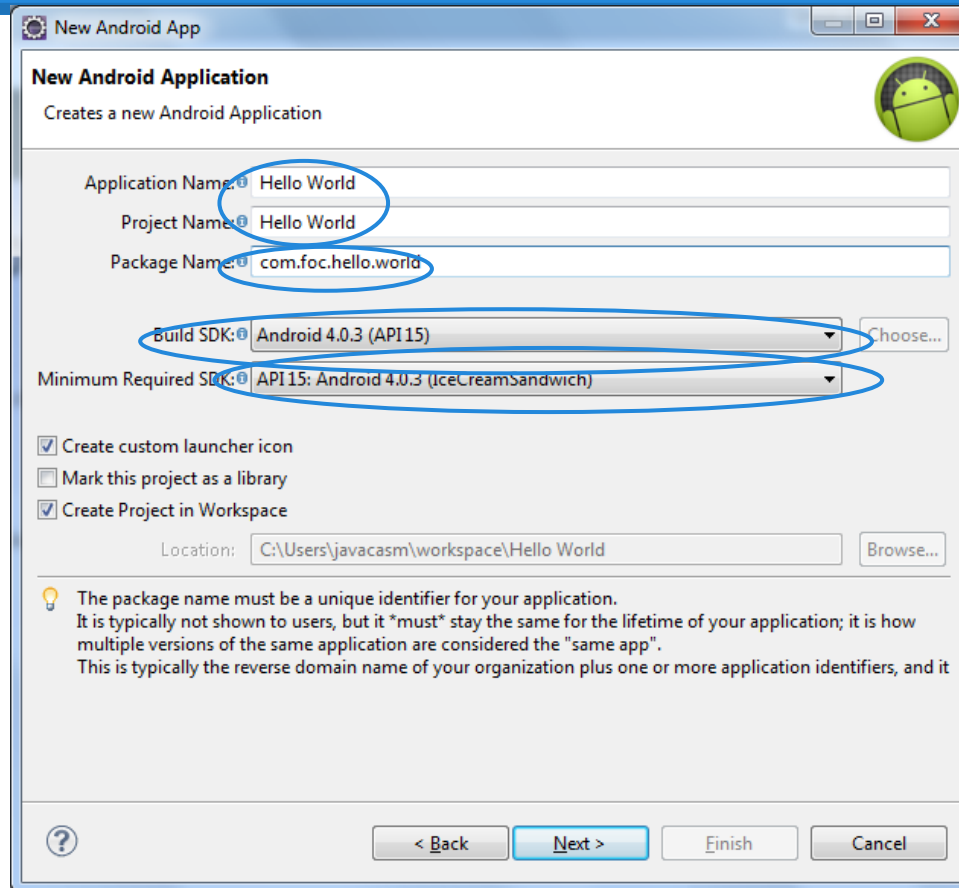
Programación en Android: SDKs



En cualquier momento podemos añadir más versiones



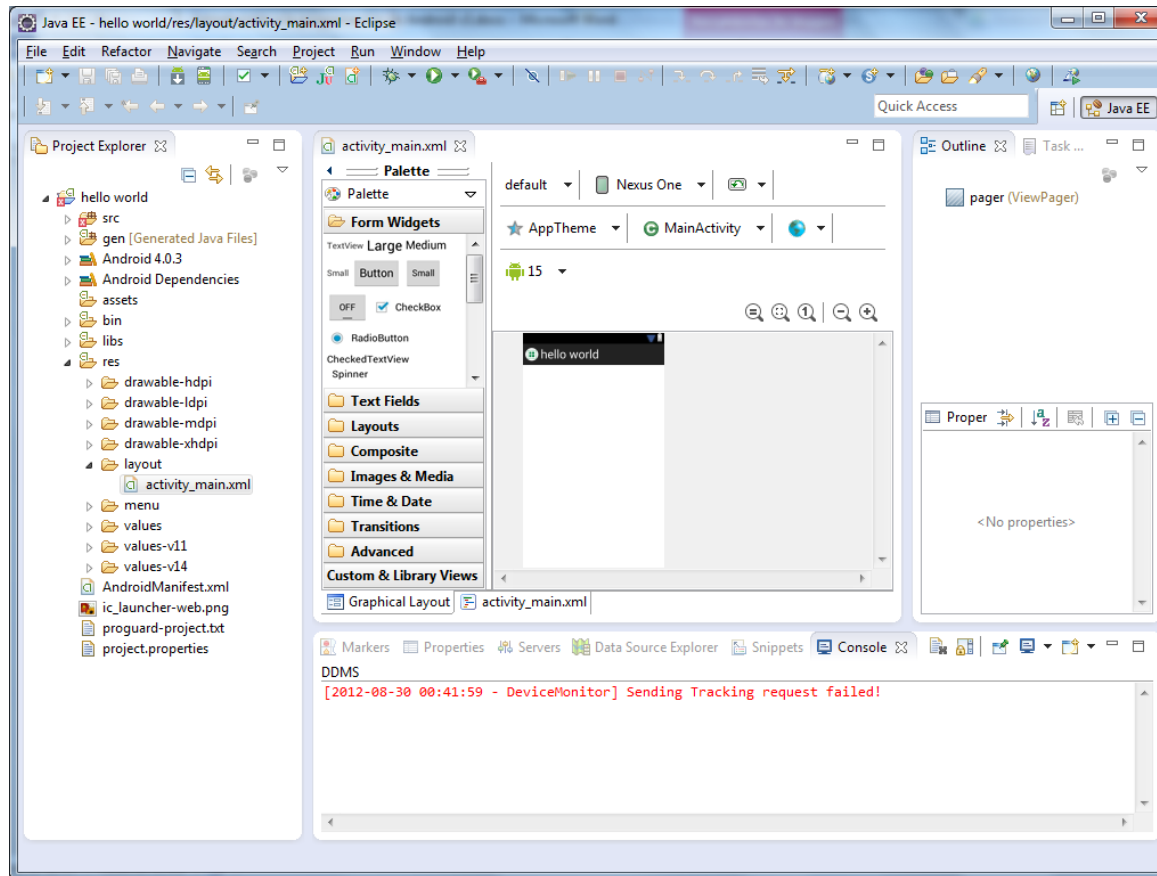
Programación en Android: Primeros pasos



Ahora asignamos las propiedades fundamentales del proyecto



Programación en Android: Primeros pasos



Si todo ha ido correctamente, tendremos cargado nuestro nuevo proyecto



Programación en Android: Arquitectura

Framework de aplicación

- **Views** (vistas): se utilizan para construir una aplicación, incluyendo lists (listas), grids (rejillas), text box (cajas de texto), buttons (botones), e incluso un navegador web embebido
- **Content Providers** (proveedores de contenido) que permiten a las aplicaciones acceder a datos de otras aplicaciones (como los contactos), o compartir sus propios datos
- **Resource Manager** (administrador de recursos), facilitar el acceso a los recursos no son de código tales como cadenas localizadas, gráficos y archivos de diseño
- **Notification Manager** (Administrador de notificaciones) que permite a todas las aplicaciones mostrar alertas personalizadas en la barra de estado
- **Activity Manager** (gestor de actividad) que gestiona el ciclo de vida de las aplicaciones y proporciona una navegación común



Programación en Android: Arquitectura

- **Activities:** manejan la interfaz de usuario y la pantalla del smartphone. Se van añadiendo a una cola LIFO
- **Services:** manejan los procesamiento en segundo plano.
- **Broadcast receivers:** manejan la comunicación entre sus aplicaciones.
- **Content providers:** manejan los datos y todo lo relacionado con la gestión de datos.
- **Intent:** contiene un mensaje que se envía entre los diferentes módulos.



Programación en Android: Arquitectura

Fichero Manifest.xml

El fichero manifest sirve para muchos más propósitos que simplemente definir los componentes de nuestra aplicación. La siguiente lista resume las partes relevantes de un fichero manifest en el contexto del desarrollo de juegos:

- La versión de nuestra aplicación tal y como se muestra y utiliza en el Market de Android
- La versión de Android en la que nuestra aplicación puede funcionar
- Perfiles de hardware que nuestra aplicación requiere (ie: multitáctil, resoluciones de pantalla específicas o soporte para OpenGL ES 2.0)
- Permisos para usar componentes específicos, tales como escritura en la tarjeta SD o acceso a la pila de red.



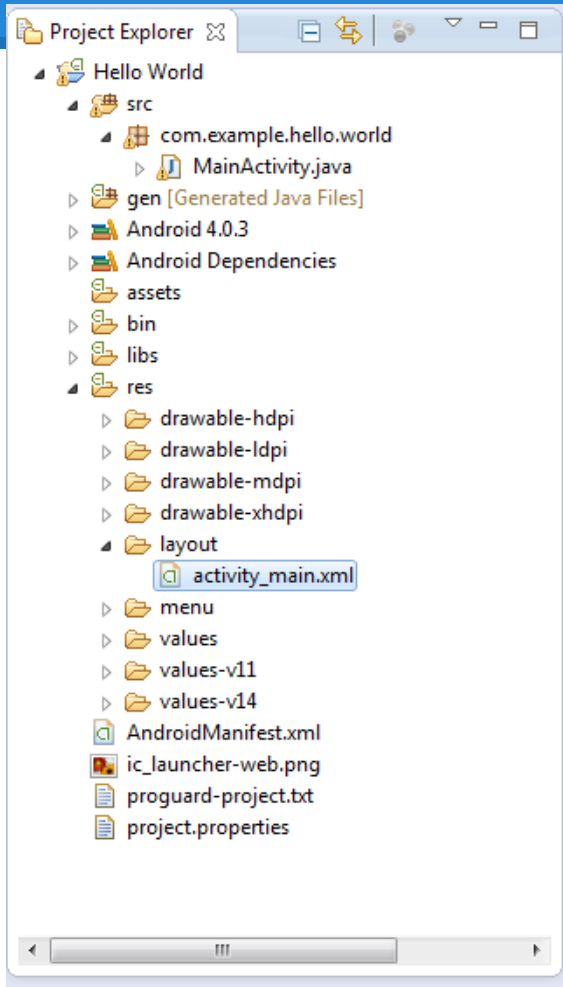
```
graph TD; Start([Activity starts]) --> onCreate[onCreate()]; onCreate --> onStart[onStart()]; onStart --> onResume[onResume()]; onResume --> Running([Activity is running]); Running -- "Another activity comes in front of the activity" --> onPause[onPause()]; onPause -- "The activity comes to the foreground" --> onResume; onPause -- "The activity is no longer visible" --> onStop[onStop()]; onStop -- "The activity comes to the foreground" --> onRestart[onRestart()]; onRestart --> onStart; onStop -- "User navigates back to the activity" --> onCreate; onStop -- "Other applications need memory" --> Killed([Process is killed]); onStop --> onDestroy[onDestroy()]; onDestroy --> ShutDown([Activity is shut down]);
```

The flowchart illustrates the lifecycle of an Android Activity. It begins with 'Activity starts', leading to 'onCreate()', then 'onStart()', and 'onResume()'. The activity then enters a 'running' state. From 'running', it can transition to 'onPause()' if 'Another activity comes in front of the activity'. From 'onPause()', it can return to 'onResume()' if 'The activity comes to the foreground', or proceed to 'onStop()' if 'The activity is no longer visible'. From 'onStop()', it can return to 'onRestart()' if 'The activity comes to the foreground', which then leads back to 'onStart()'. Alternatively, 'onStop()' can lead to 'onCreate()' if 'User navigates back to the activity', or to 'onDestroy()' if 'Other applications need memory'. Finally, 'onDestroy()' leads to 'Activity is shut down'.



Programación en Android

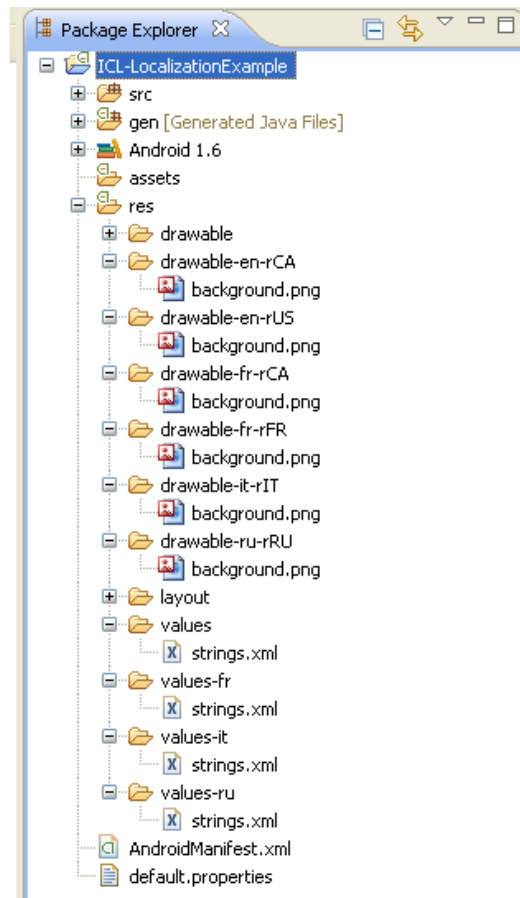
Estructura de proyecto



- **AndroidManifest.xml** describe su aplicación. Define de qué actividades y servicios está compuesto, qué versiones mínima y objetivo de Android se suponen que van a ejecutarse y qué permisos necesitan
- **project.properties** contiene varias configuraciones para construir el sistema. No debemos tocar aquí ya que el plug-in de ADT lo modifica cuando es necesario
- **src/** contiene todos los ficheros fuente Java. Note que el paquete tiene el mismo nombre que el nombre del proyecto Android
- **gen/** contiene los ficheros fuente Java generados por el sistema de construcción de Android. Estos se generan automáticamente en algunos casos. El más importante es el que se puede observar en la imagen, el fichero **R.java**, y la clase **R**. Esta clase **R** contendrá en todo momento una serie de constantes con los ID de todos los recursos de la aplicación incluidos en la carpeta **/res/**, de forma que podamos acceder fácilmente a estos recursos desde nuestro código a través de este dato. Así, por ejemplo, la constante **R.drawable.icon** contendrá el ID de la imagen "icon.png" contenida en la carpeta **/res/drawable/**
- **assets/** es donde se almacenan los ficheros que nuestra aplicación necesite (i.e. ficheros de configuración, ficheros de audio, etc.). Estos ficheros se empaquetarán con la aplicación Android
- **res/** contiene recursos que la aplicación necesite como iconos, cadenas de texto para internacionalización, apariencia del UI definidos vía XML. También se empaquetarán con la aplicación Android
- **Android 4.0.3** indica que se está generando contra una versión de Android 4.0.3. Esto es normalmente una dependencia en la forma de un JAR estándar que contiene las clases de la API Android 4.0.3.
- El Explorador de Paquetes contiene otro directorio llamado **bin/** que aloja el código compilado que se necesita para el dispositivo o el emulador. Al igual que la carpeta **gen/** no nos preocupamos mucho de ellos.



Programación en Android: Recursos - Traducciones



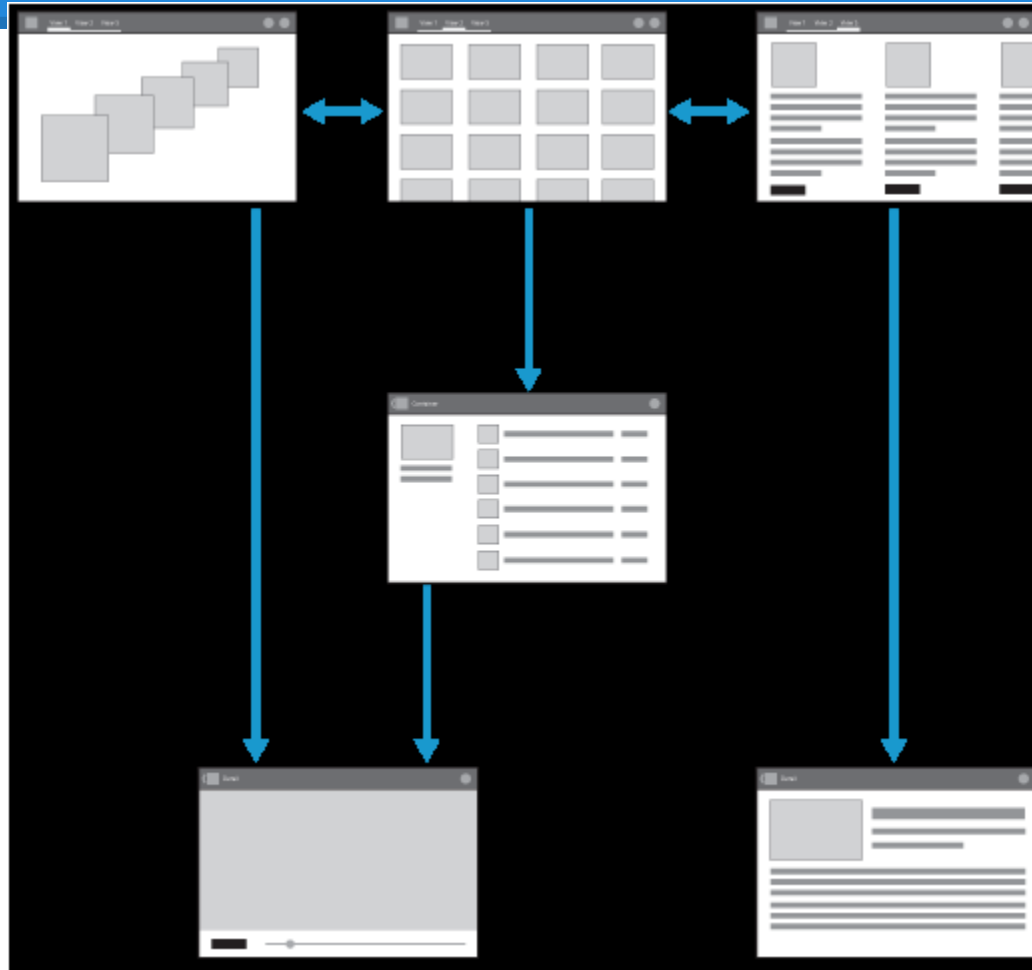
Del mismo modo en el caso de que tengamos imágenes distintas dependiendo del idioma duplicaremos la carpeta para cada idioma con los diferentes contenidos.

En el ejemplo:

Italian	drawable-it-rIT/background.png
French	drawable-fr-rFR/background.png
French (Canada)	drawable-fr-rCA/background.png
English (Canada)	drawable-en-rCA/background.png
Russian	drawable-ru-rRU/background.png
US English	drawable-en-rUS/background.png
Default (Earth image)	drawable/background.png



Programación en Android: Diferentes tipos de layouts

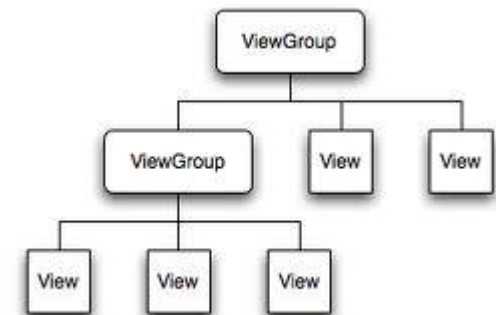


Programación en Android: Aplicaciones

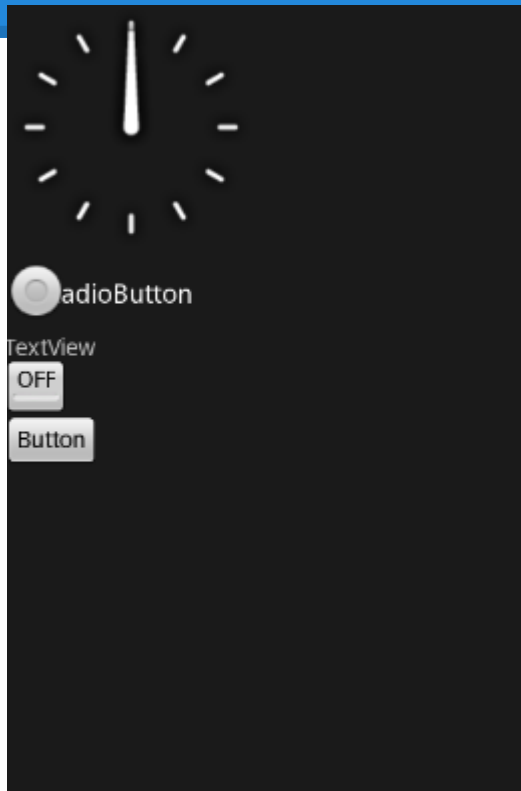
Los objetos visuales de Android se agrupan en **layouts** sobre los que colocamos los elementos gráficos

Todos heredan directa o indirectamente de la clase View.

- **View:** clase básica para la creación de todos los componentes usados en la interfaz. View ocupa un área rectangular en la pantalla y es el responsable de dibujar los componentes y manejar los eventos que definamos sobre ellos. Es la clase base de todos los controles que vienen incluidas en la plataforma Android, como son botones, campos de textos, etc
- **ViewGroup.** Hereda directamente de View y se usa para, agrupar y controlar la lista de Views y de otros ViewGroups. Es la clase base para los Layouts, mediante los cuales podemos construir una estructura con nuestros Views.



Programación en Android: LinearLayout



Los componentes se colocan uno a continuación del otro, siguiendo la dirección elegida. Podemos asignar pesos (layout_weight) y definir gravity

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <AnalogClock
        android:id="@+id/analogClock1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <RadioButton
        android:id="@+id/radioButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="RadioButton" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView" />

    <ToggleButton
        android:id="@+id/toggleButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="ToggleButton" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button" />

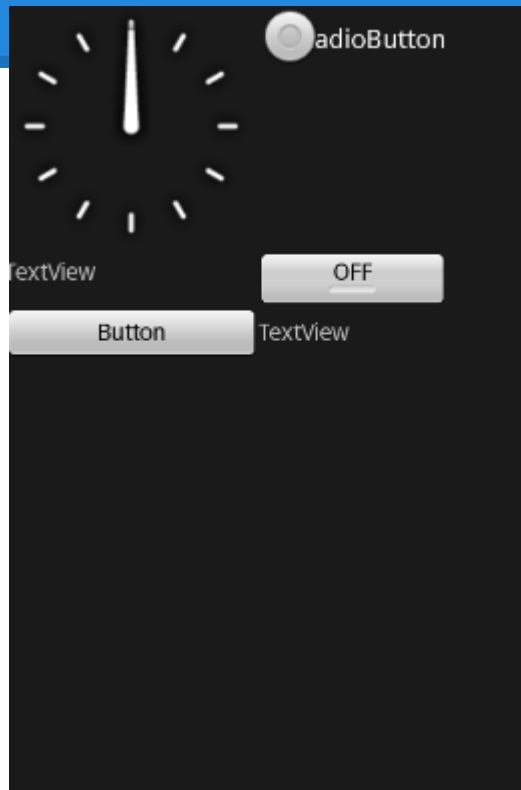
</LinearLayout>
```

android:gravity posiciona los elementos en la vista
android:layout_gravity posiciona la vista con respecto a su padre.



Programación en Android

Tablelayout



```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <TableRow >

        <AnalogClock
            android:id="@+id/analogClock1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

        <RadioButton
            android:id="@+id/radioButton1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="RadioButton" />

    </TableRow>

    <TableRow >

        <TextView
            android:id="@+id/textView1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="TextView" />

        <ToggleButton
            android:id="@+id/toggleButton1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="ToggleButton" />

    </TableRow>

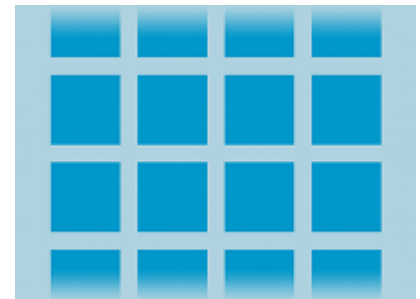
    <TableRow >

        <Button
            android:id="@+id/button1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Button" />

        <TextView
            android:id="@+id/textView2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="TextView" />

    </TableRow>

</TableLayout>
```

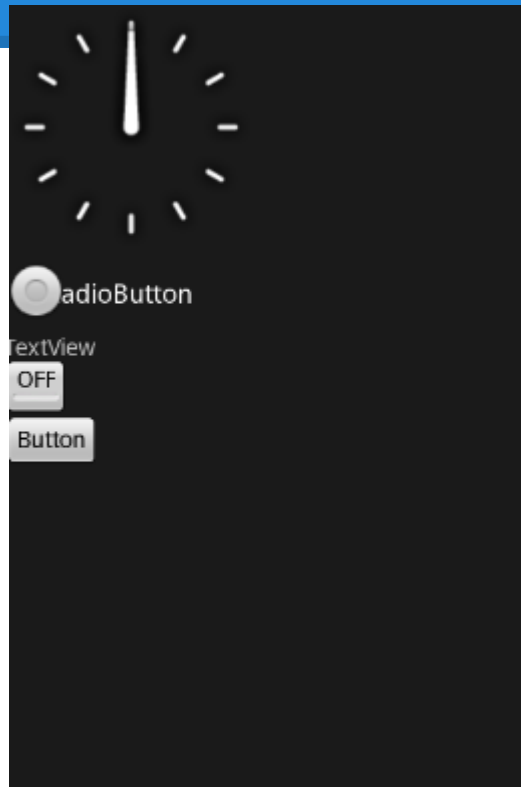


Los componentes se distribuyen
en filas utilizando la etiqueta
TableRow



Programación en Android

RelativeLayout

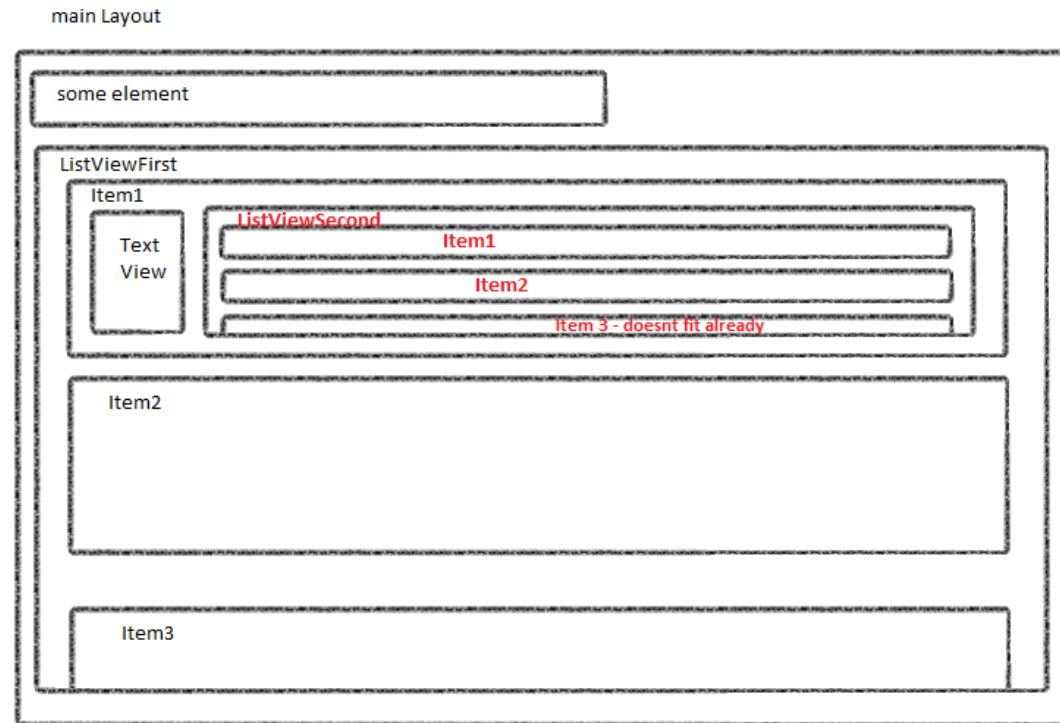


Tipo	Propiedades
Posición relativa a otro control	<code>android:layout_above</code> <code>android:layout_below</code> <code>android:layout_toLeftOf</code> <code>android:layout_toRightOf</code> <code>android:layout_alignLeft</code> <code>android:layout_alignRight</code> <code>android:layout_alignTop</code> <code>android:layout_alignBottom</code> <code>android:layout_alignBaseline</code>
Posición relativa al layout padre	<code>android:layout_alignParentLeft</code> <code>android:layout_alignParentRight</code> <code>android:layout_alignParentTop</code> <code>android:layout_alignParentBottom</code> <code>android:layout_centerHorizontal</code> <code>android:layout_centerVertical</code> <code>android:layout_centerInParent</code>
Opciones de margen (también disponibles para el resto de layouts)	<code>android:layout_margin</code> <code>android:layout_marginBottom</code> <code>android:layout_marginTop</code> <code>android:layout_marginLeft</code> <code>android:layout_marginRight</code>
Opciones de espaciado o padding (también disponibles para el resto de layouts)	<code>android:padding</code> <code>android:paddingBottom</code> <code>android:paddingTop</code> <code>android:paddingLeft</code> <code>android:paddingRight</code>



Programación en Android: Casos reales

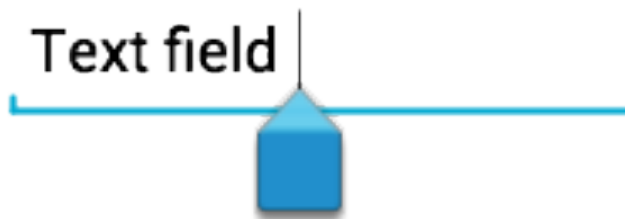
En aplicaciones reales utilizaremos layouts anidados, hasta conseguir el diseño deseado



No se deben de anidar demasiados niveles porque se pueden producir problemas de rendimiento



Programación en Android: Views



[Detalles](#)



Programación en Android

Views - TextView

TextView

Text field

Este control se usa para mostrar textos al usuario. Se le pueden asignar diferentes atributos (a parte de las ya conocidas como **android:layout_width** y **android:layout_height** para definir el ancho y la altura) como pueden ser, tamaño de fuente (**android:textSize**, recordad usad la medida **sp** en este caso), color de fuente (**android:textColor**), un fondo personalizado (**android:background**), etc¹. Para definir un texto se usa el atributo **android:text**.

```
< TextView
    android:id = "@+id/text_view"
    android:layout_width = "fill_parent"
    android:layout_height = "wrap_content"
    android:text = "@string/some_text" />
```

Si queremos cambiar este texto desde código, tenemos que hacerlo de la siguiente manera:

```
TextView textView = (TextView) findViewById(R.id. text_view );
textView.setText("Estoy cambiando el texto del TextView por por este");
```

Se puede ver que hemos usado el método **findViewById()**. Mediante este método accedemos al id del elemento que hemos definido en el XML. Todos los id que creamos, se añaden automáticamente al fichero R.java como ya vimos, por lo que para acceder a un id tenemos que hacer referencia a dicha clase mediante *R.id.nombre_id*. Y si lo que queremos es acceder al texto que contiene, haremos lo siguiente:

```
String texto = textView.getText().toString();
```



Programación en Android

Views - Button

Button



Es el botón más básico que podemos utilizar en Android. Al heredar directamente del TextView, se le pueden asignar cualquiera de los atributos anteriormente comentados. Un ejemplo de uso es el siguiente:

```
<Button  
  android:id="@+id/button"  
  android:layout_width="wrap_content"  
  android:layout_height="wrap_content"  
  android:text="@string/text_button" />
```



Programación en Android

Views - Togglebutton

Togglebutton



Es un tipo de botón que puede permanecer en dos estados, pulsado o no pulsado, por lo tanto debemos asignarle un texto para cuando esté pulsado, y otro para cuando no esté pulsado, mediante **android:textOn** y **android:textOff**. Hereda indirectamente de `TextView`, por lo que podemos usar los atributos ya comentados anteriormente. Veamos un ejemplo de uso:

Como vemos, por defecto aparecerá pulsado (ya que hemos puesto el valor del atributo **android:checked** a `true`), por lo que se verá el texto ON.

```
<ToggleButton
    android:id="@+id/toggle_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:checked="true"
    android:textOff="@string/text_off"
    android:textOn="@string/text_on" />
```

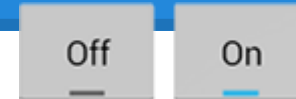


Programación en Android

Views - Togglebutton

Togglebutton

Podemos asignar distintas imágenes a los dos estados usando un **selector** que colocaremos en `res/drawable/toggle_style.xml`



```
<?xml version="1.0" encoding="UTF-8"?>
<selector
xmlns:android="http://schemas.android.com/apk/res/android">
<item android:state_checked="false"
android:drawable="@drawable/toggle_off" />
<item android:state_checked="true"
android:drawable="@drawable/toggle_on" />
</selector>
```

Asignamos los atributos al botón

```
<ToggleButton android:id="@+id/BtnBoton4"
android:textOn="ON"
android:textOff="OFF"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:background="@drawable/toggle_style"/>
```



Programación en Android

Views - CheckBox



CheckBox

Marcar o desmarcar opciones. Al heredar indirectamente de TextView, se le pueden asignar cualquiera de los atributos de éste ya comentados. Junto al texto aparecerá la casilla del CheckBox.

```
<CheckBox  
android:id="@+id/checkbox"  
android:layout_width = "wrap_content"  
android:layout_height = "wrap_content"  
android:text = "@string/check_me" />
```

```
if (checkBox.isChecked()) {  
    checkBox.setChecked(false);  
}
```

Evento: `onCheckedChangeListener`.



Programación en Android

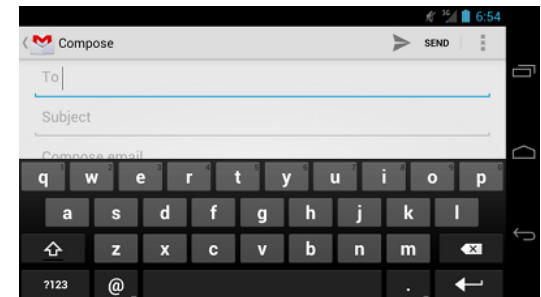
View - EditText

EditText

Este control sirve para la edición de texto. Es el que se debe usar para la introducción y edición de texto por parte del usuario.

Podemos usar los mismos métodos que con el TextView para cambiar el texto o para recuperarlo, ya que hereda directamente de él.

```
<EditText  
    android:id = "@+id/edit_text"  
    android:layout_width = "fill_parent"  
    android:layout_height = "wrap_content"  
    android:inputType = "text" />
```



Programación en Android

View - EditText

Spanned (spandable)

```
//Creamos un nuevo objeto de tipo Editable
Editable str = Editable.Factory.getInstance().newEditable("Esto es un simulacro.");
//Marcamos como fuente negrita la palabra "simulacro"
str.setSpan(new StyleSpan(android.graphics.Typeface.BOLD), 11, 19 ,
Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);

//Obtiene el texto del control con etiquetas de formato HTML
String aux2 = Html.toHtml(txtTexto.getText());

//Asigna texto con formato HTML
txtTexto.setText( Html.fromHtml("<p>Esto es un <b>simulacro</b>.</p>"), BufferType.
SPANNABLE);
```



Programación en Android Views - Spinners

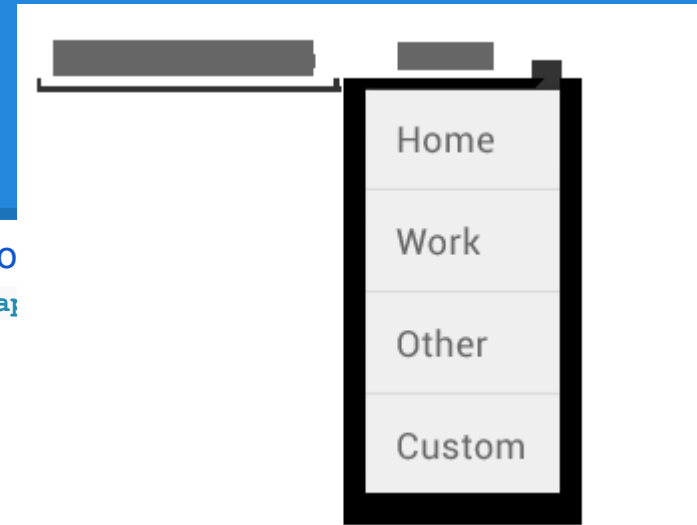
Spinners

Es una forma rápida de permitir que el usuario seleccione una opción. Proporcionaremos estas opciones por medio de un `SpinnerAdapter` (por ejemplo un array de strings en recursos)

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="planets_array">
        <item>Mercury</item>
        <item>Venus</item>
        <item>Earth</item>
        <item>Mars</item>
        <item>Jupiter</item>
        <item>Saturn</item>
        <item>Uranus</item>
        <item>Neptune</item>
    </string-array>
</resources>
```

El código para asignar las opciones

```
Spinner spinner = (Spinner) findViewById(R.id.spinner);
ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,
    R.array.planets_array, android.R.layout.simple_spinner_item);
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spinner.setAdapter(adapter);
```



Programación en Android Views - Spinners

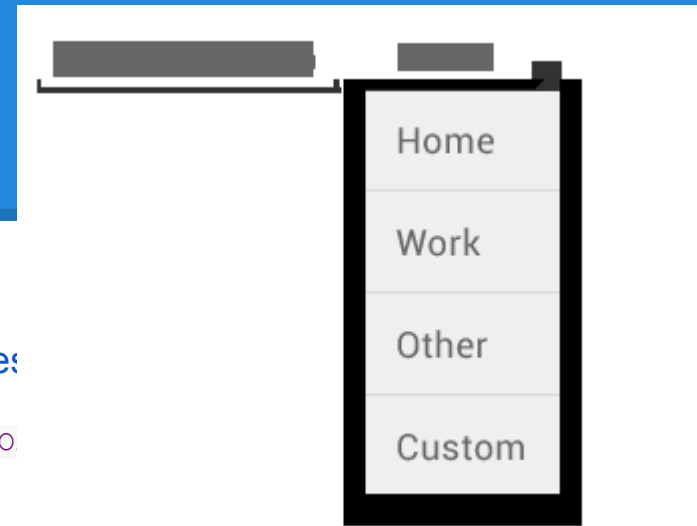
Spinners

Para recuperar la selección del usuario usaremos los siguientes:

```
public class SpinnerActivity extends Activity implements OnItemSelectedListener {  
    ...  
  
    public void onItemSelected (AdapterView<?> parent, View view,  
        int pos, long id) {  
        // An item was selected. You can retrieve the selected item using  
        // parent.getItemAtPosition(pos)  
    }  
  
    public void onNothingSelected (AdapterView<?> parent) {  
        // Another interface callback  
    }  
}
```

Y añadiremos al onCreate

```
Spinner spinner = (Spinner) findViewById(R.id.spinner);  
spinner.setOnItemSelectedListener (this);
```



Programación en Android: Enlaces de iconos gratuitos

<http://www.androidicons.com>



arrow down.png



arrow up.png



copy.png



cut.png



options.png



paste.png



redo.png



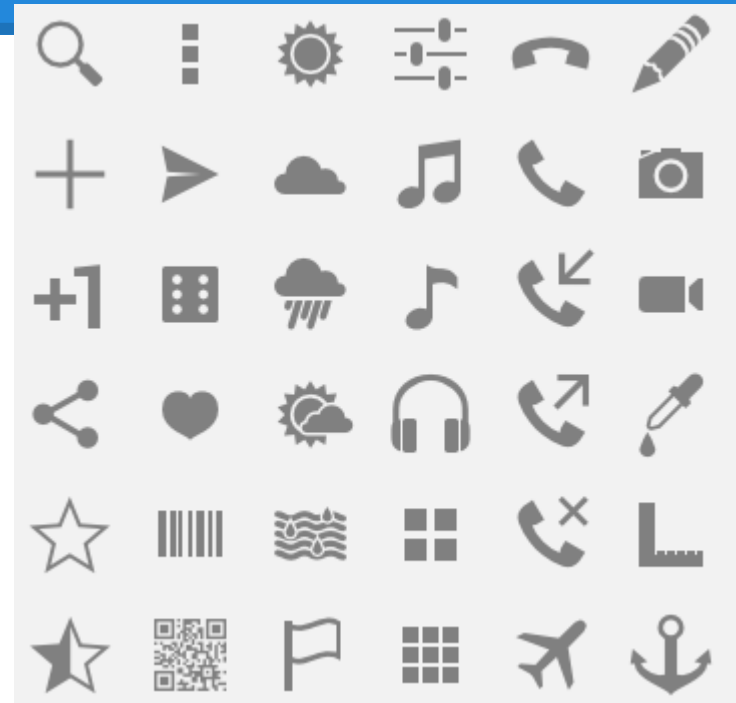
undo.png



zoom in.png



zoom out.png



http://www.glyfx.com/products/free_android2.html



Programación en Android: Enlaces de iconos gratuitos

Android Asset Studio

<http://android-ui-utils.googlecode.com/hg/asset-studio/dist/index.html>

Explorador de iconos de android

<http://androiddrawableexplorer.appspot.com/>

Guia de estilo de Android

http://developer.android.com/guide/practices/ui_guidelines/index.html

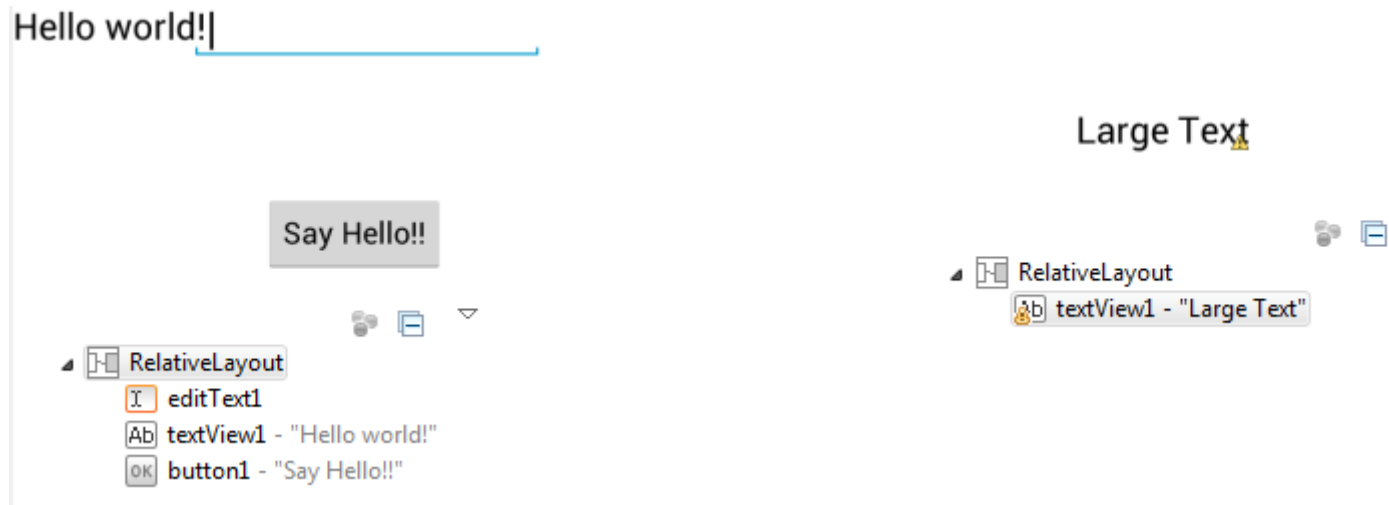
App: R.android



Programación en Android

Aplicaciones - Segunda activity

Interface de usuario



Programación en Android

Aplicaciones - Segunda activity

Segunda activity

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="15dp"
        android:text="Large Text"
        android:textAppearance="?android:attr/textAppearanceLarge" />

</RelativeLayout>
```



Programación en Android

Aplicaciones - Segunda activity

```
<RelativeLayout xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" xmlns:android="http://schemas.android.com/apk/res/android">

    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:ems="10"
        android:inputType="text" >

        <requestFocus />
    </EditText>

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/editText1"
        android:layout_alignBottom="@+id/editText1"
        android:text="@string/hello_world"
        android:textAppearance="?android:attr/textAppearanceLarge"
        tools:context=".MainActivity" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editText1"
        android:layout_marginLeft="34dp"
        android:layout_marginTop="80dp"
        android:layout_toRightOf="@+id/textView1"
        android:text="@string/button_hello_world" />

</RelativeLayout>
```



Programación en Android

Aplicaciones - Código primera activity

```
public class MainActivity extends Activity {  
  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        final EditText txtNombre=(EditText)findViewById(R.id.editText1);  
  
        final Button btnHola=(Button)findViewById(R.id.button1);  
  
        btnHola.setOnClickListener(new OnClickListener() {  
  
            public void onClick(View v) {  
                Intent intent = new Intent(MainActivity.this, SecondActivity.class);  
                Bundle bundle =new Bundle();  
                bundle.putString("NOMBRE", txtNombre.getText().toString());  
  
                intent.putExtras(bundle);  
                startActivity(intent);  
            }  
        });  
    }  
  
    @Override  
    public boolean onCreateOptionsMenu(Menu menu) {  
        getMenuInflater().inflate(R.menu.activity_main, menu);  
        return true;  
    }  
}
```



Programación en Android

Aplicaciones - Código segunda activity

Código segunda activity

```
public class SecondActivity extends Activity {  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_second);  
  
        TextView txtMensaje=(TextView)findViewById(R.id.textView1);  
  
        Bundle bundle=getIntent().getExtras();  
  
        String sNombre=bundle.getString("NOMBRE");  
        txtMensaje.setText("Hola "+sNombre);  
    }  
  
    @Override  
    public boolean onCreateOptionsMenu(Menu menu) {  
        getMenuInflater().inflate(R.menu.activity_second, menu);  
        return true;  
    }  
}
```



Programación en Android

Aplicaciones - Manifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.test2activities"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="10"
        android:targetSdkVersion="15" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".SecondActivity"
            android:label="@string/title_activity_second" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```



Programación en Android

Aplicaciones - Strings.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.test2activities"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="10"
        android:targetSdkVersion="15" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".SecondActivity"
            android:label="@string/title_activity_second" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```



Programación en Android: Recursos

ANDROID developer.android.com

Muy bueno http://www.sgoliver.net/blog/?page_id=3011

Avanzado <http://www.limecreativelabs.com/curso-gratuito-de-desarrollo-para-android/>

Avanzado(En) <http://www.vogella.com/android.html>

MiriadaX http://miriadax.net/es/web/android_programacion

