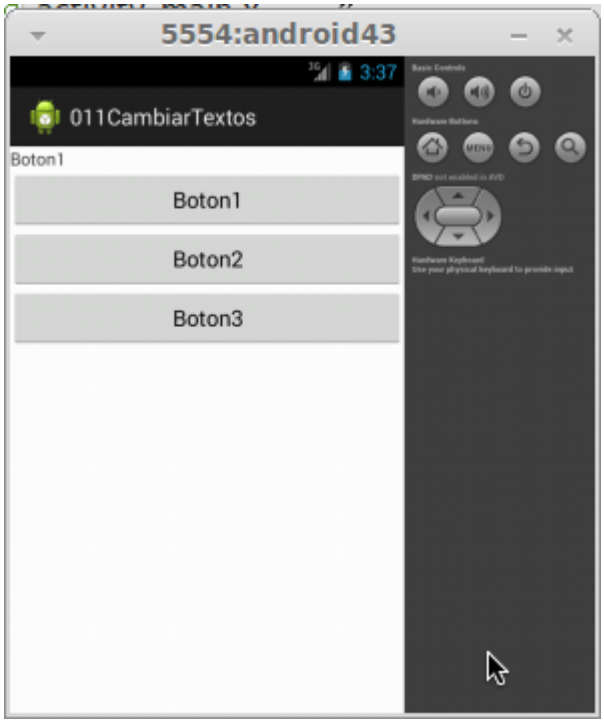
Android es una plataforma de desarrollo que empieza a tener el sobrenombre de INFINITA en cuanto a las cosas que se pueden hacer, conjunto de APIs soportadas etc . Sin embargo al ser tan grande muchas veces nos centramos en aprender los conceptos avanzados y nos olvidamos un poco de lo más elemental. Vamos a ver un ejemplo sencillo que contiene 3 botones.



Vamos a ver el código del Layout :

```
<LinearLayout android:id="@+id/zona"
 xmlns:android="http://schemas.android.com/apk/res/android"
 android:layout_width="fill_parent"
 android:layout_height="fill_parent" android:orientation="vertical" >

 <TextView
```

```
 android:id="@+id/texto_1"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 />

 <Button
 android:id="@+id/boton_1"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:text="@string/boton_1" />

 <Button
 android:id="@+id/boton_2"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:text="@string/boton_2" />

 <Button
 android:id="@+id/boton_3"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:text="@string/boton_3" />

</LinearLayout>
```

El programa que tenemos que construir es sencillo simplemente cuando pulsamos cada uno de los botones el texto contenido en el boton se mostrará en la caja de texto . Vamos a ver el código de la actividad:

```
package com.arquitecturajava.cambiartextos;

import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

import com.example.cambiartextos.R;

public class MainActivity extends Activity implements OnClickListener
{

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

Button boton1 = (Button) findViewById(R.id.boton_1);
boton1.setOnClickListener(this);

Button boton2 = (Button) findViewById(R.id.boton_2);
boton2.setOnClickListener(this);

Button boton3 = (Button) findViewById(R.id.boton_3);
boton3.setOnClickListener(this);

}
```

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is
present.
getMenuInflater().inflate(R.menu.main, menu);
return true;
}

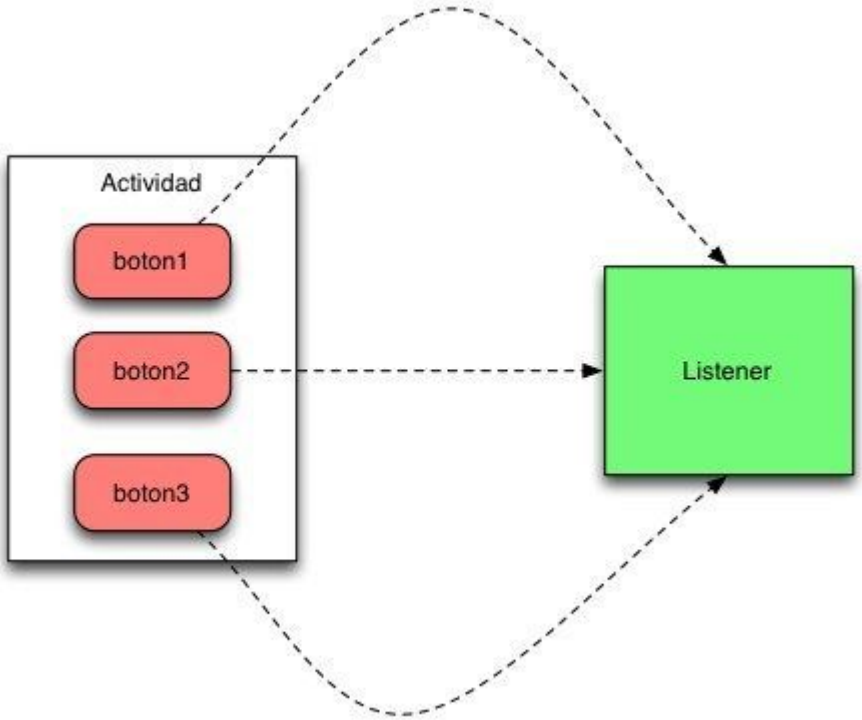
@Override
public void onClick(View v) {

TextView texto = (TextView) findViewById(R.id.texto_1);
Button boton = (Button) v;
texto.setText(boton.getText());

}

}
```

Cada uno de los botones registra el mismo método como listener (Método onClick) ya que lo único que tiene que hacer es imprimir el texto del botón por pantalla.



Revisando nuestro código nos podemos dar cuenta que tenemos un problema . Por cada botón que tenemos debemos utilizar el método findViewById.

```
<pre>@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

Button boton1 = (Button) findViewById(R.id.boton_1);
boton1.setOnClickListener(this);

Button boton2 = (Button) findViewById(R.id.boton_2);
boton2.setOnClickListener(this);
```

```
Button boton3 = (Button) findViewById(R.id.boton_3);
boton3.setOnClickListener(this);

}</pre>
```

Es evidente que cuantos mas botones tengamos mas estaremos repitiendo código ya que todos los eventos hacen lo mismo . ¿Como podemos solventar este problema? .

Android TAG

En este caso podemos usar una de las características básicas de los controles y menos conocidas los Android TAG que nos permiten marcar varios controles con la misma "Etiqueta" en nuestro caso la denominaremos "grupoBotones". Veamos como queda modificado el Layout.

```
<LinearLayout android:id="@+id/zona"
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent" android:orientation="vertical" >

<TextView
android:id="@+id/texto_1"
android:layout_width="match_parent"
android:layout_height="wrap_content"
/>

<Button
android:id="@+id/boton_1"
android:layout_width="match_parent"
android:layout_height="wrap_content"
```

```
android:tag="grupoBotones"
android:text="@string/boton_1" />

<Button
android:id="@+id/boton_2"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:tag="grupoBotones"
android:text="@string/boton_2" />

<Button
android:id="@+id/boton_3"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:tag="grupoBotones"
android:text="@string/boton_3" />

</LinearLayout>
```

Realizada esta operación podemos modificar el método onCreate de nuestra actividad para que genere los listener para todos aquellos controles que compartan el mismo TAG .

```
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

LinearLayout grupo = (LinearLayout) findViewById(R.id.zona);
Log.v("mensaje", grupo.getChildCount() + "");
```

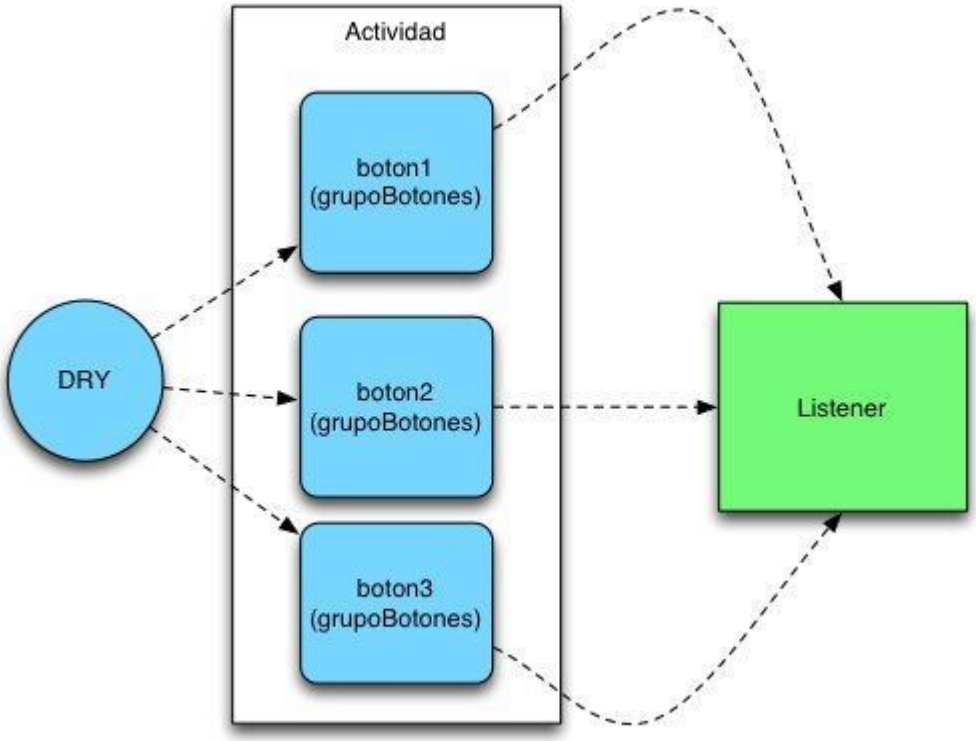
```
Button boton = null;
for (int i = 0; i < grupo.getChildCount(); i++) {

View vista = grupo.getChildAt(i);
if (vista.getTag() != null &&
vista.getTag().toString().equals("grupoBotones")) {

boton = (Button) vista;
boton.setOnClickListener(this);
}
}

}
```

De esta forma habremos cumplido con el principio DRY utilizando Android TAG ya que podremos añadir los botones que queramos sin repetir código.


Actividad
boton1
(grupoBotones)
boton2
(grupoBotones)
boton3
(grupoBotones)
DRY
Listener