



# El Componente Button

## CONCEPTOS.

Un botón es un componente gráfico que usualmente permite la interacción del usuario con las aplicaciones, presionándolo o con clic. Se representa con un widget dentro de una plantilla.

La jerarquía de la clase Button:

```
public class Button
    extends TextView
    java.lang.Object
    android.view.View
    android.widget.TextView
    android.widget.Button
```

Subclases de Button:

AppCompatButton, CompoundButton

Subclases indirectas de Button:

AppCompatCheckBox, AppCompatRadioButton, CheckBox, RadioButton, Switch, SwitchCompat, ToggleButton



- El ejemplo básico de un botón es el siguiente:

- El código Java del botón en la actividad MiActividad.java:

```
public class MainActivity extends Activity {
    protected void onCreate(Bundle b) {
        super.onCreate(b);
        setContentView(R.layout.content_layout_id);
        final Button jbn = (Button) findViewById(R.id.button_id);
        bn.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // Desarrollo de alguna acción.
            }
        });
    }
}
```

- El código del activity\_main.xml:

```
<Button
    android:id="@+id/bn_id"
    android:text="Aceptar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

- Los constructores públicos de Button son:

```
Button(Context context)
Button(Context context, AttributeSet attrs)
Button(Context context, AttributeSet attrs, int defStyleAttr)
Button(Context context, AttributeSet attrs, int defStyleAttr, int defStyleRes)
```

- Un método público:

```
CharSequence getAccessibilityClassName ()
```



- Algunos atributos de Button:

android:id	Genera automáticamente una constante de identificación en la clase R para este botón.
android:text	El texto del botón. Se puede asignar con el prefijo "@string/nom_variable".
android:layout_height	Define la altura del botón.
android:layout_width	Define lo ancho del botón.
android:layout_margin	Indica el espacio entre el botón y su padre, en este caso el Layout.
android:padding	Indica el espacio entre el texto o imágenes en el interior del botón.
android:gravity	Indica alineación del texto. Con "center" centra el contenido vertical y horizontalmente.
android:drawable	Imagen del botón. Con android:drawableBottom, la imagen se pone bajo el texto.
android:textSize	Indica el tamaño del texto en unidades dp. Si no se indica, el tamaño es el standard.
android:backgroud	Define el color de fondo del botón. Se puede usar un recurso de colors.xml.
android:onClick	Se utiliza a partir de la API 6. Se le asigna un nombre de método y se le invoca en el Java.
android:background	El color de fondo del botón.
android:enabled	Con "true" o "false" se habilitan o deshabilitan los eventos del usuario.
android:textColor	Determina el color del texto en el botón.

## EJEMPLO 1.

**Paso 1.** Capturar el siguiente código en el MainActivity.java:

```
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener; // -1
import android.widget.*;

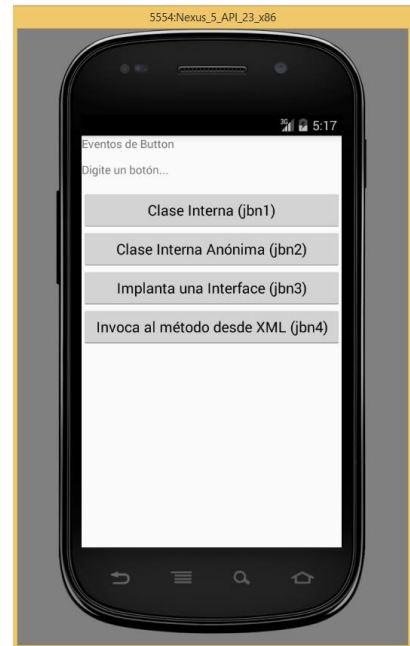
public class MainActivity extends Activity implements OnClickListener{ // -2
    Button jbn1, jbn2, jbn3, jbn4;
    public void onCreate(Bundle b) {
        super.onCreate(b);
        setContentView(R.layout.activity_main);
        jbn1 = (Button) findViewById(R.id.xbn1); // MÉTODO 1
        jbn1.setOnClickListener(bn1Listener); // -3
        jbn2 = (Button) findViewById(R.id.xbn2); // MÉTODO 2
        jbn2.setOnClickListener(new OnClickListener() {
            public void onClick(View v) {
                showMessage("Botón digitado: xbn2\nUtiliza: new OnClickListener{}");
            }
        });
        jbn3 = (Button) findViewById(R.id.xbn3); // MÉTODO 3
        jbn3.setOnClickListener(this);
    }
    private OnClickListener bn1Listener = new OnClickListener() { // desde el MÉTODO 1
        public void onClick(View v) {
            showMessage("Botón digitado: xbn1\nUtiliza: clase btn1Listener");
        }
    };
    public void onClick(View v) { // -4 y -5
        showMessage("Botón digitado: xbn3\nUtiliza: implements OnClickListener.");
    }
    public void xbn4DesdeXML(View v) { // MÉTODO 4
        showMessage("Boton digitado: xbn4\nInvoca al método desde el XML.");
    }
    private void showMessage(String msg){
        Toast toast = Toast.makeText(this, msg, Toast.LENGTH_SHORT);
        toast.show();
    }
}
```



Los comentarios de los métodos indican cuatro formas de invocar al método `onClick(View v)`. Los comentarios numéricos indican los cinco pasos básicos para que un botón realice una función determinada.

## Paso 2. Capturar el siguiente código en el `activity_main.xml`:

```
<?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" >
    <TextView
        android:text="Eventos de Button\n\nDigite un botón...\n"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <Button
        android:text="Clase Interna (jbn1)"
        android:id="@+id/xbn1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <Button
        android:text="Clase Interna Anónima (jbn2)"
        android:id="@+id/xbn2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <Button
        android:text="Implanta una Interface (jbn3)"
        android:id="@+id/xbn3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <Button
        android:text="Invoca al método desde XML (jbn4)"
        android:id="@+id/xbn4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="xbn4DesdeXML" />
</LinearLayout>
```



## Paso 3. La ejecución debe ser similar a la imagen anterior.

- El botón `ImageButton`.

La imagen se incluye con el atributo `android:src`. La imagen se puede colocar, por ejemplo, en la carpeta `/res/drawable/imagen.png`. Para referenciarla, en el XML se indica con `android:src="@drawable/imagen"`. Con el atributo `android:contentDescription` se puede asignar una descripción textual.

Algunos atributos de `ImageButton`:

<code>android:adjustViewBounds</code>	Si es "true" se ajustan los límites y se preserva la relación de aspecto.
<code>android:baselineAlignBottom</code>	Si es "true", la imagen se alinearán con la orilla del botón.
<code>android:cropToPadding</code>	Si es "true", la imagen se recortará para ajustarse con un espaciado.
<code>android:baseline</code>	El offset de la línea base dentro del botón.
<code>android:src</code>	Inserta una imagen en el interior del botón.

- El botón `ToggleButton`.

Este botón posee dos estados, **ON/OFF**. Posee un texto para `android:textOn` y otro para `android:textOff`.

**EJEMPLO 2.****Paso 1.** Capturar el siguiente código en el MainActivity.java:

```

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener; // 1
import android.widget.*;
public class MainActivity extends Activity implements OnClickListener{ // 2
    ImageButton    jib1;
    ToggleButton    jtb1;
    public void onCreate(Bundle b) {
        super.onCreate(b);
        setContentView(R.layout.activity_main);
        jib1 = (ImageButton) findViewById(R.id.xib1); jib1.setOnClickListener(this); // 3
        jtb1 = (ToggleButton) findViewById(R.id.xtb1); jtb1.setOnClickListener(this);
    }
    public void onClick(View v) { // 4
        if(v.getId()==R.id.xib1) // 5
            showMessage("Botón: ImageButton");
        else {
            if (jtb1.isChecked()) showMessage("Botón: ToggleButton en ON");
            else showMessage("Botón: ToggleButton en OFF");
        }
    }
    private void showMessage(String s){
        Toast t = Toast.makeText(this, s, Toast.LENGTH_SHORT);
        t.show();
    }
}

```

Los comentarios numéricos indican los cinco pasos básicos para que un botón realice una función determinada.

**Paso 2.** Capturar el siguiente código en el activity\_main.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <TextView
        android:text="Botón ImageButton:"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <ImageButton
        android:id="@+id/xib1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@mipmap/ic_launcher" />
    <TextView
        android:text="\nBotón ToggleButton:"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
    <ToggleButton
        android:id="@+id/xtb1"
        android:textOn="ON"
        android:textOff="OFF"

```





```

        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>

```

**Paso 3.** La ejecución del proyecto debe mostrar una imagen similar a la anterior.

- El siguiente es un botón que puede incluir imagen y texto, a la vez:

```

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="@drawable/imagen1"
    android:drawableTop="@drawable/imagen2"
    android:textColor="#ffffff"
    android:id="@+id/btnImgTxt"
    android:paddingTop="32sp"
    android:drawablePadding="-15sp"
    android:text="Aceptar" />

```

O también:

```

<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:drawableLeft="@drawable/imagen"
    android:gravity="left|center_vertical" />

```

## **EJERCICIOS**

### **EJERCICIO 1.**

Diseñar una aplicación que al digitar un botón se reproduzca un sonido. Crear una carpeta `raw` en el repositorio y guardar allí el sonido, por ejemplo `sonido.wav`. Utilizar el código siguiente.

En el `MainActivity.java`:

```

import android.media.MediaPlayer;
import android.os.Bundle;
import android.app.Activity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.*;

public class MainActivity extends Activity {
    MediaPlayer mp;
    Button      btn;
    public void onCreate(Bundle b) {
        super.onCreate(b);
        setContentView(R.layout.activity_main);
        btn= (Button) findViewById(R.id.btnSonido);
        btn.setOnClickListener(new OnClickListener() {
            public void onClick(View v) {
                sonido.start();
            }
        });
        mp = MediaPlayer.create(MainActivity.this, R.raw.sonido);
        mp.start();
    }
    public void onDestroy() {
        mp.stop();
        super.onDestroy();
    }
}

```



En el `activity_main.xml`:

```
<Button
    android:id="@+id/btnSonido"
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:layout_gravity="center_horizontal"
    android:text="@string/btn_text" />
```

## EJERCICIO 2.

Diseñar una aplicación que muestre un botón flotante `FloatingActionButton`. Cambiar la imagen interior del botón, el mensaje del `TextView`, el mensaje del `Toast` y la posición del botón flotante. Utilizar el código siguiente.

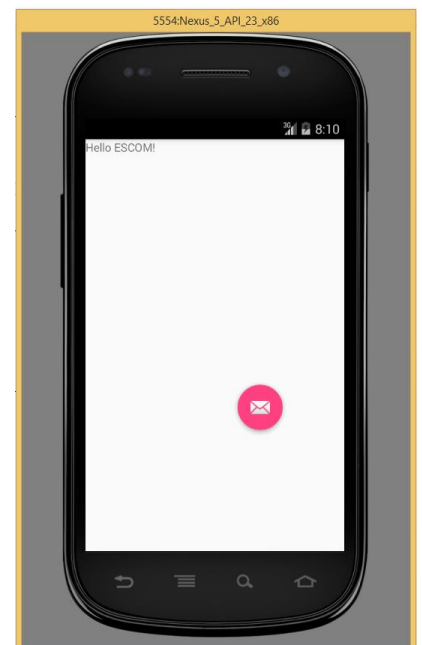
En el `MainActivity.java`:

```
import android.app.Activity;
import android.os.Bundle;
import android.view.*;
import android.widget.Toast;
import android.support.design.widget.FloatingActionButton;

public class MainActivity extends Activity {
    FloatingActionButton jfab;
    protected void onCreate(Bundle b) {
        super.onCreate(b);
        setContentView(R.layout.activity_main);
        jfab = (FloatingActionButton) findViewById(R.id.xfab);
        jfab.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                Toast.makeText(MainActivity.this, "Soy un botón Flotante!",
                    Toast.LENGTH_LONG).show();
            }
        });
    }
}
```

En el `activity_main.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <android.support.design.widget.FloatingActionButton
        android:id="@+id/xfab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom|end"
        android:src="@android:drawable/ic_dialog_email"
        android:layout_marginRight="64dp"
        android:layout_marginEnd="64dp"
        android:layout_marginBottom="131dp"
        android:layout_alignParentBottom="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hola ESCOM!" />
</RelativeLayout>
```

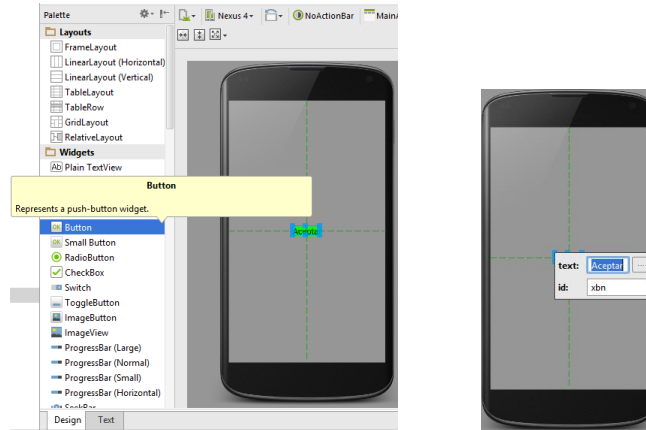




## Programación gráfica de botones.

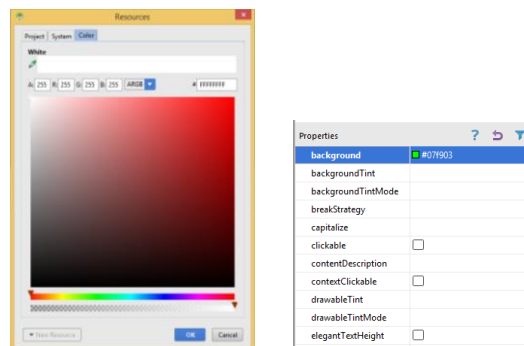
**Paso 1.** Crear un nuevo proyecto y abrir el archivo `activity_main.xml`.

Seleccionar la pestaña **Design**. Localizar en las opciones de la paleta el componente `Button`. Seleccionarlo, arrastrarlo y soltarlo sobre la plantilla. Si se digita doble clic sobre la imagen del botón, se muestran las opciones para el cambio de texto y el identificador.



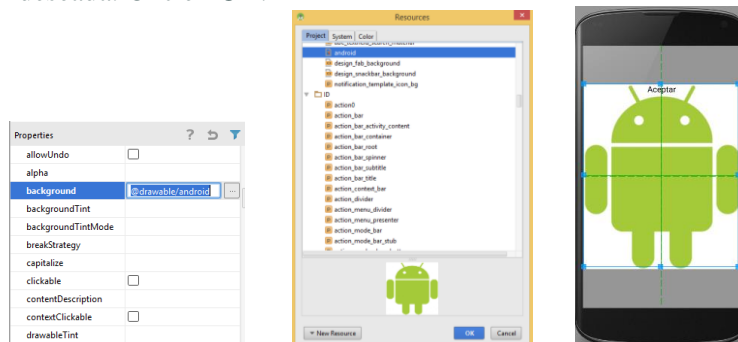
**Paso 2.** Propiedades de los componentes.

Para cambiar el color, seleccionar el botón y localizar en las opciones de las propiedades la opción **background**. Clic en el botón `background` `#ffffff`, a la derecha de la propiedad, para seleccionar el color deseado en la ventana **Resources**. Clic en OK.



**Paso 3.** Agregar una imagen al botón.

Primero, guardar la imagen en la carpeta **drawable**, por ejemplo `res/drawable/imagen.png`. Luego, clic en el botón `background` `#ffffff`, a la derecha de la propiedad `background`, para mostrar la ventana **Resources** y en la pestaña **Project** seleccionar la imagen deseada. Clic en OK.





El código del `activity_main.xml` debe ser similar al siguiente:

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

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Aceptar"
        android:id="@+id/xbn"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:textAlignment="center"
        android:textSize="20sp"
        android:background="@drawable/android" />
</RelativeLayout>
```