**1.- activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>

<!—

Los widgets que estén dentro de éste contenedor basarán su posición en relación con los otros elementos.

android:layout\_height: Especifica la altura básica de la vista.

android:layout\_width Especifica el ancho básico de la vista.

android:background: Cambia el color o fondo del layout.

android:orientation: Cambia la orientación del layout.

tools:context: Declara a qué actividad está asociada esta disposición de forma predeterminada.

-->

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app=<http://schemas.android.com/apk/res-auto>

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="@android:color/holo\_green\_light"

android:orientation="vertical"

tools:context="com.juangabrielgomila.counter.MainActivity">

<!--

Los TextView se utilizan como medio de salida, es decir, para mostrar un determinado texto al usuario. Al igual que en el caso de los botones, el texto del control se establece mediante la propiedad android:text.

A parte de esta propiedad, las etiquetas que definen su tamaño son las mínimas que debemos definir.

Para generalizar:

android:layout\_alignParentTop: Indica al widget que su borde superior deberá estar alineado con el borde superior del contenedor.

android:layout\_alignParentEnd: Si es verdadero, hace que el borde final de esta vista coincida con el borde final del elemento primario. -->

android:layout\_alignParentStart Si es verdadero, hace que el borde inicial de esta vista coincida con el borde de inicio del elemento primario.

-->

<TextView

android:id="@+id/text\_view"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentEnd="true"

android:layout\_alignParentStart="true"

android:layout\_alignParentTop="true"

android:text="@string/text\_view\_0"

android:textAlignment="center"

android:textColor="@color/colorPrimary"

android:textSize="30sp" />

<!--

El Button es el botón más básico que podemos utilizar y normalmente contiene un simple texto. En el código se definió un botón con el texto asignando su propiedad android:text. Además de esta propiedad podríamos utilizar muchas otras como el color de fondo (android:background), estilo de fuente (android:typeface), color de fuente (android:textcolor), tamaño de fuente (android:textSize), etc.

-->

<Button

android:id="@+id/btnAdd"

style="@style/Widget.AppCompat.Button.Borderless"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentStart="true"

android:layout\_below="@+id/text\_view"

android:layout\_marginStart="35dp"

android:layout\_marginTop="24dp"

android:text="@string/btn\_add"

android:textColor="@color/colorAccent"

android:textSize="24sp" />

<!—

El Button es el botón más básico que podemos utilizar y normalmente contiene un simple texto. En el código se definió un botón con el texto asignando su propiedad android:text.

Además de esta propiedad podríamos utilizar muchas otras como el color de fondo (android:background), estilo de fuente (android:typeface), color de fuente (android:textcolor), tamaño de fuente (android:textSize), etc.

-->

<Button

android:id="@+id/btnGrow"

style="@style/Widget.AppCompat.Button.Borderless"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignStart="@+id/btnAdd"

android:layout\_below="@+id/btnAdd"

android:layout\_marginTop="34dp"

android:text="@string/btn\_grow"

android:textColor="@color/colorAccent"

android:textSize="24sp" />

<!—

El Button (botón) es el botón más básico que podemos utilizar y normalmente contiene un simple texto. En el código se definió un botón con el texto asignando su propiedad android:text. Además de esta propiedad podríamos utilizar muchas otras como el color de fondo (android:background), estilo de fuente (android:typeface), color de fuente (android:textcolor), tamaño de fuente (android:textSize), etc.

-->

<Button

android:id="@+id/btnHide"

style="@style/Widget.AppCompat.Button.Borderless"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignStart="@+id/btnGrow"

android:layout\_centerVertical="true"

android:text="@string/btn\_hide"

android:textColor="@color/colorAccent"

android:textSize="24sp" />

<!—

El Button (botón) es el botón más básico que podemos utilizar y normalmente contiene un simple texto. En el código se definió un botón con el texto asignando su propiedad android:text. Además de esta propiedad podríamos utilizar muchas otras como el color de fondo (android:background), estilo de fuente (android:typeface), color de fuente (android:textcolor), tamaño de fuente (android:textSize), etc.

-->

<Button

android:id="@+id/btnTake"

style="@style/Widget.AppCompat.Button.Borderless"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_above="@+id/btnGrow"

android:layout\_alignParentEnd="true"

android:layout\_marginEnd="26dp"

android:text="@string/btn\_take"

android:textColor="@color/colorAccent"

android:textSize="24sp" />

<!—

El Button (botón) es el botón más básico que podemos utilizar y normalmente contiene un simple texto. En el código se definió un botón con el texto asignando su propiedad android:text. Además de esta propiedad podríamos utilizar muchas otras como el color de fondo (android:background), estilo de fuente (android:typeface), color de fuente (android:textcolor), tamaño de fuente (android:textSize), etc.

-->

<Button

android:id="@+id/btnShrink"

style="@style/Widget.AppCompat.Button.Borderless"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignStart="@+id/btnTake"

android:layout\_alignTop="@+id/btnGrow"

android:text="@string/btn\_shrink"

android:textColor="@color/colorAccent"

android:textSize="24sp" />

<!—

El Button es el botón más básico que podemos utilizar y normalmente contiene un simple texto. En el código se definió un botón con el texto asignando su propiedad android:text. Además de esta propiedad podríamos utilizar muchas otras como el color de fondo (android:background), estilo de fuente (android:typeface), color de fuente (android:textcolor), tamaño de fuente (android:textSize), etc.

-->

<Button

android:id="@+id/btnReset"

style="@style/Widget.AppCompat.Button.Borderless"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignStart="@+id/btnShrink"

android:layout\_alignTop="@+id/btnHide"

android:text="@string/btn\_reset"

android:textColor="@color/colorAccent"

android:textSize="18sp" />

</RelativeLayout>

**2.- strings.xml**

<!—

Un recurso de string ofrece strings de texto para tu aplicación con formato y estilo de texto opcionales.

-->

<resources>

<!--

Este es el nodo raíz.

-->

<!--

A partir de aquí se llaman las Cadenas creadas junto con sus botones correspondientes.

-->

<string name="app\_name">Counter</string>

<string name="text\_view\_0">0</string>

<string name="btn\_add">+</string>

<string name="btn\_grow">A+</string>

<string name="btn\_hide">Ocultar</string>

<string name="btn\_take">-</string>

<string name="btn\_shrink">A-</string>

<string name="btn\_reset">Reset</string>

</resources>

<!--

Aquí finalizan los Resources.

-->

**3.- MainActivity.java**

package com.juangabrielgomila.counter; /\* Nombre asignado al paquete en AS \*/

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.TextView;

public class MainActivity extends AppCompatActivity implements View.OnClickListener { /\*\* Se hereda de la clase principal, conservando ciertos atributos\*/

//Variable para sostener el valor entero que vamos a mostrar

private int value = 0;

//Variables para los 6 botones y la text view

private TextView textView;

private Button btnAdd, btnTake, btnGrow, btnShrink, btnHide, btnReset;

@Override /\*\* Indica que una declaración de método está destinada a anular una declaración de método en un supertipo. \*/

protected void onCreate(Bundle savedInstanceState) { /\*\* Se llama cuando se crea por primera vez la actividad. Aquí es donde se debe hacer toda la configuración estática normal: crear vistas, vincular datos a listas, etc. \*/

super.onCreate(savedInstanceState); /\*\* Cuando anulamos un método, tenemos la opción de reemplazar completamente el método en nuestra clase, o de extender el método existente de la clase padre. Al llamar super.onCreate(savedInstanceState);, le dice a la VM Dalvik que ejecute su código además del código existente en onCreate () de la clase padre. Si omite esta línea, solo se ejecuta su código. El código existente se ignora por completo. \*/

setContentView(R.layout.activity\_main); /\*\* Se usa con un recurso de diseño que define su UI\*/

textView = (TextView) findViewById(R.id.text\_view); /\*\* Muestra texto al usuario y, opcionalmente, le permite editarlo mediante programación. \*/

btnAdd = (Button) findViewById(R.id.btnAdd); /\*\*Se llama al objeto reconociendolo por su id antes asignado. \*/

btnTake = (Button) findViewById(R.id.btnTake); /\*\*Se llama al objeto reconociendolo por su id antes asignado. \*/

btnGrow = (Button) findViewById(R.id.btnGrow); /\*\*Se llama al objeto reconociendolo por su id antes asignado. \*/

btnShrink = (Button) findViewById(R.id.btnShrink); /\*\*Se llama al objeto reconociendolo por su id antes asignado. \*/

btnHide = (Button) findViewById(R.id.btnHide); /\*\*Se llama al objeto reconociendolo por su id antes asignado. \*/

btnReset = (Button) findViewById(R.id.btnReset); /\*\*Se llama al objeto reconociendolo por su id antes asignado. \*/

btnAdd.setOnClickListener(this); /\*\* El sistema ejecuta el código en el que escribe onClick(View)después de que el usuario presiona el botón. \*/

btnTake.setOnClickListener(this); /\*\* Registra una devolución de llamada que se invocará cuando se haga clic y se mantenga presionada esta vista. En esta vista se puede hacer clic cuantas veces se desee. \*/

btnGrow.setOnClickListener(this); /\*\* Registra una devolución de llamada que se invocará cuando se haga clic y se mantenga presionada esta vista. En esta vista se puede hacer clic cuantas veces se desee. \*/

btnShrink.setOnClickListener(this); /\*\* Registra una devolución de llamada que se invocará cuando se haga clic y se mantenga presionada esta vista. En esta vista se puede hacer clic cuantas veces se desee. \*/

btnHide.setOnClickListener(this); /\*\* Registra una devolución de llamada que se invocará cuando se haga clic y se mantenga presionada esta vista. En esta vista se puede hacer clic cuantas veces se desee. \*/

btnReset.setOnClickListener(this); /\*\* Registra una devolución de llamada que se invocará cuando se haga clic y se mantenga presionada esta vista. En esta vista se puede hacer clic cuantas veces se desee. \*/

}

@Override /\*\* Indica que una declaración de método está destinada a anular una declaración de método en un supertipo. \*/

public void onClick(View view) { /\*\* Este método se llama cuando el usuario toca el elemento (en el modo táctil), o selecciona el elemento con las teclas de navegación o la bola de seguimiento y presiona la tecla “Entrar” adecuada o la bola de seguimiento.\*/

Log.i("Main Activity", "onClick: "+view.getId());

float scale;

switch (view.getId()){

case R.id.btnAdd:

value++;

textView.setText(""+value);

break;

case R.id.btnTake:

value--;

textView.setText(""+value);

break;

case R.id.btnReset:

value = 0;

textView.setText(""+value);

break;

case R.id.btnGrow:

scale = textView.getTextScaleX();

scale = scale +1;

textView.setTextScaleX(scale);

break;

case R.id.btnShrink:

scale = textView.getTextScaleX();

scale = scale -1;

textView.setTextScaleX(scale);

break;

case R.id.btnHide:

if (textView.getVisibility() == View.VISIBLE){

/\*\*En este caso, la vista está visible y debe ocultarla....\*/

textView.setVisibility(View.INVISIBLE);

btnHide.setText("Mostrar");

}else {

/\*\*En este caso, la vista está invisible, ahora debe mostrarla...\*/

textView.setVisibility(View.VISIBLE);

btnHide.setText("Ocultar");

}

break;

}

}

}

**4.- SomeClass.java**

package com.juangabrielgomila.counter; /\*\* Se nombra a la paquetería \*/

import android.app.Application;

import android.util.Log;

import android.widget.Toast;

/\*\*

\* Created by JuanGabriel on 6/11/17.

\*/

public class SomeClass { /\*\* Se inicia la clase principal y se nombra, esta es la principal forma de organización de los ficheros de código \*/

int day; /\*\* Declaramos una variable tipo Int (Entero) llamada "day" \*/

public void aMethod(){ /\*\* El método público se ejecuta pero no devuelve ningún tipo de dato, utilizaremos el valor ‘void’ (vacío) para el tipo de dato a devolver. \*/

final String stringNice = "S"; /\*\* Localiza Cadenas en la vista \*/

class AnonymousClass{ /\*\* Se crea una clase independiente\*/

int holaQueAse; /\*\* Se declara una variable del tipo Int \*/

public AnonymousClass(){ /\*\* Se determina qué la clase será Pública \*/

}

private void sayHello(){ /\*\* Se inicializa un nuevo método \*/

Log.i("HOLA", "sayHello: "+stringNice+holaQueAse+day); /\*\* Envía un "INFO", mensaje de registro. \*/

}

}

}

public class InnerClass{ /\*\* Inicializa la clase \*/

int age; /\*\* Declara varriable Age \*/

String name; /\*\* Age será tipo Cadena \*/

public InnerClass(){

}

public int addOne(){ /\*\* Se regresará el valor dado con +1 de resultado \*/

return age+1;

}

public void innerMethod(){

day = day+1;

}

}

}

/\*public class OtherClass{

}\*/

**5.- AndroidManifest.xml**

<?xml version="1.0" encoding="utf-8"?>

<!—

El archivo de manifiesto proporciona información esencial sobre tu aplicación al sistema Android, información que el sistema debe tener para poder ejecutar el código de la app.

-->

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.juangabrielgomila.counter">

<!--

android:allowBackup: Es para permitir que la aplicación participe en la infraestructura de respaldo o restauración.

Si este atributo se establece en falso, nunca se realizará una copia de seguridad o restauración de la aplicación, ni

siquiera mediante una copia de seguridad del sistema completo que, de lo contrario, haría que todos los datos de la aplicación se

guardaran mediante adb. El valor predeterminado de este atributo es verdadero.

-->

<application

android:allowBackup="true"

android:icon="@mipmap/ic\_launcher"

android:label="@string/app\_name"

android:roundIcon="@mipmap/ic\_launcher\_round"

android:supportsRtl="true"

android:theme="@style/AppTheme">

<!--

activity android:name Nombre de la clase que implementa a la actividad; una subclase de Activity. El valor del atributo debe ser un

nombre de clase completamente calificado.

-->

<activity android:name=".MainActivity">

<!--

Especifica los tipos de intentos a los que una actividad, servicio o receptor de difusión puede responder. Un filtro de

intención declara las capacidades de su componente principal: qué puede hacer una actividad o servicio y qué tipos de transmisiones

puede manejar un receptor.

-->

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

</application>

</manifest>

<!--

Concluye manifiesto.

-->

6 app

<?xml version="1.0" encoding="utf-8"?>

<!—

El archivo de manifiesto proporciona información esencial sobre tu aplicación al sistema Android, información que el sistema debe tener para poder ejecutar el código de la app.

-->

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.juangabrielgomila.counter">

<!--

android:allowBackup: Es para permitir que la aplicación participe en la infraestructura de respaldo o restauración.

Si este atributo se establece en falso, nunca se realizará una copia de seguridad o restauración de la aplicación, ni

siquiera mediante una copia de seguridad del sistema completo que, de lo contrario, haría que todos los datos de la aplicación se

guardaran mediante adb. El valor predeterminado de este atributo es verdadero.

-->

<application

android:allowBackup="true"

android:icon="@mipmap/ic\_launcher"

android:label="@string/app\_name"

android:roundIcon="@mipmap/ic\_launcher\_round"

android:supportsRtl="true"

android:theme="@style/AppTheme">

<!--

activity android:name Nombre de la clase que implementa a la actividad; una subclase de Activity. El valor del atributo debe ser un

nombre de clase completamente calificado.

-->

<activity android:name=".MainActivity">

<!--

Especifica los tipos de intentos a los que una actividad, servicio o receptor de difusión puede responder. Un filtro de

intención declara las capacidades de su componente principal: qué puede hacer una actividad o servicio y qué tipos de transmisiones

puede manejar un receptor.

-->

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

</application>

</manifest>

<!--

Concluye manifiesto.

-->