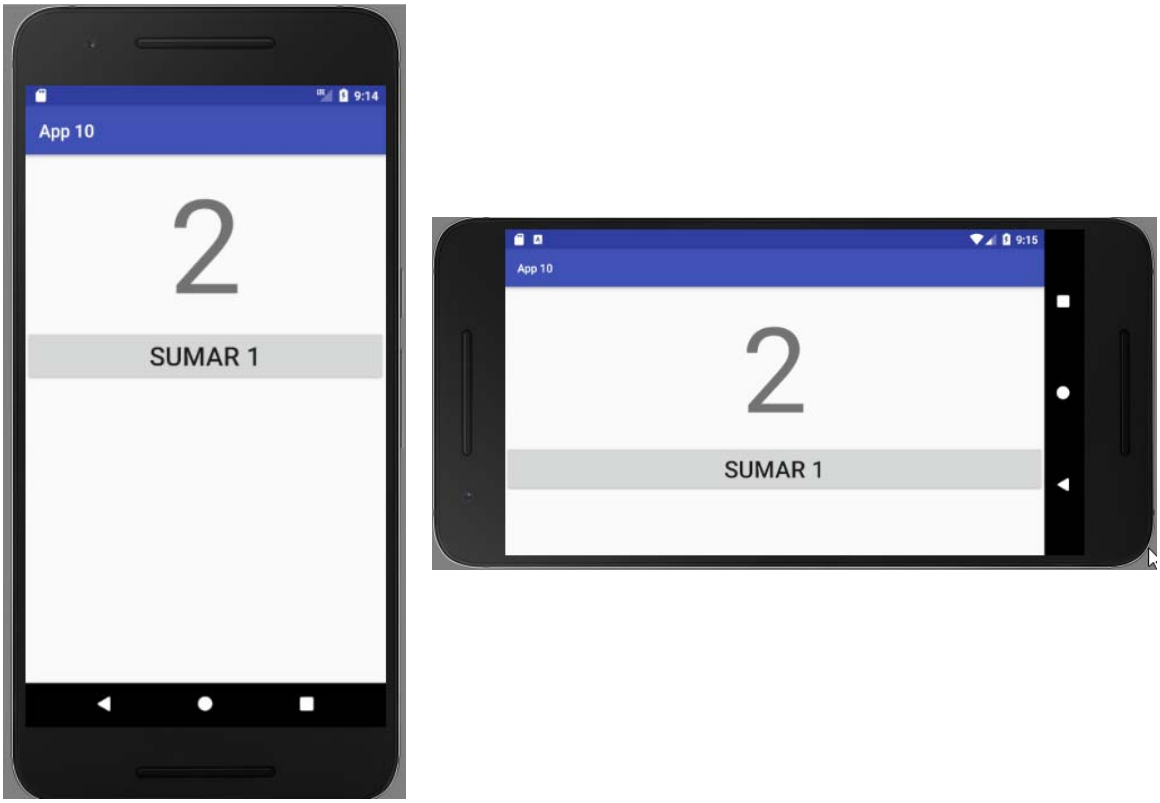
# PRACTICA 10.

## OBJETIVO GENERAL.

El alumno aplicará los métodos usados para salvar y restaurar valores de una actividad para evitar que se destruyan al crear de nuevo la instancia.




## OBJETIVOS ESPECÍFICOS:

- Utilizar LinearLayout como contenedor principal de la aplicación.
- Crear objetos Actividad:
  - Un TextView : Utilizado para desplegar la cita
  - Un Button : Utilizado para sumar una unidad al contador interno.
- Crear un evento que sume la unidad y la despliegue en la pantalla.
- Sobre escribir los métodos `onSaveInstanceState` y `onRestoreInstanceState`

## CREACION DE LA INTERFACE GRAFICA.

Usaremos el código siguiente para la interface gráfica.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.delio.app10.MainActivity">

<TextView
    android:id="@+id/txtContador"
    android:textSize="150sp"
    android:text="0"
    android:textAlignment="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<Button
    android:id="@+id/btnSumar"
    android:textSize="30sp"
    android:text="Sumar 1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />


</LinearLayout>
```

## DESARROLLO DE LA LOGICA DE APLICACIÓN.

El código inicial de nuestra actividad principal se verá:

```
private TextView txtContador;
private Button btnSumar;
private Integer contador =0;

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

    txtContador = (TextView) findViewById(R.id.txtContador);
    btnSumar    = (Button)   findViewById(R.id.btnSumar);
    btnSumar.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            sumarUno();
        }
    });
```

```
}
private void sumarUno(){
    contador = contador + 1;
    txtContador.setText(contador.toString());
}

@Override
protected void onSaveInstanceState(Bundle estado) {
    super.onSaveInstanceState(estado);

    estado.putString("contadorcadena",txtContador.getText().toString());
    estado.putInt("contadornumero", contador);

}

@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);
    contador = savedInstanceState.getInt("contadornumero");
    txtContador.setText(savedInstanceState.getString("contadorcadena"));

}
```

Explicaremos la lógica principal, el método onSaveInstanceState y onRestoreInstanceState son escuchadores que se disparan cuando se detecta algún cambio en la vista que se esta ejecutando, por tanto dentro de estos escuchadores debemos de guardar aquellos valores que deseamos que no se pierdan, por ejemplo, cuando cambiamos de vertical a horizontal, se guardan en forma automática y cuando se regresa se restablecen dichos valores guardados, claro esta, DEBEMOS DE INDICARLO en cada método respectivamente.

NOTA: Es posible restaurar valores sin el escuchador OnRestoreInstanceState mediante el siguiente código:

```
if(savedInstanceState!=null){
    contador = savedInstanceState.getInt("contadornumero");
    txtContador.setText(savedInstanceState.getString("contadorcadena"));
}
```

Este código seria puesto al final del evento onCreate.

Son dos formas de salvar valores y no perderlos, usted decide cual de los dos desea implementar en sus proyectos.