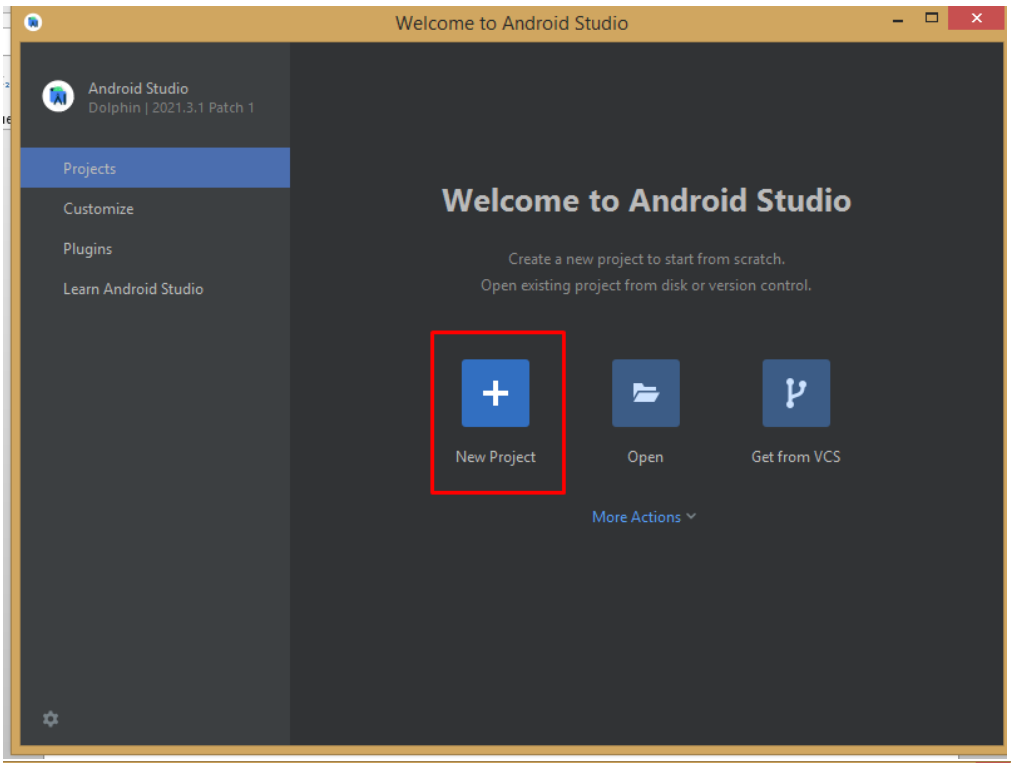
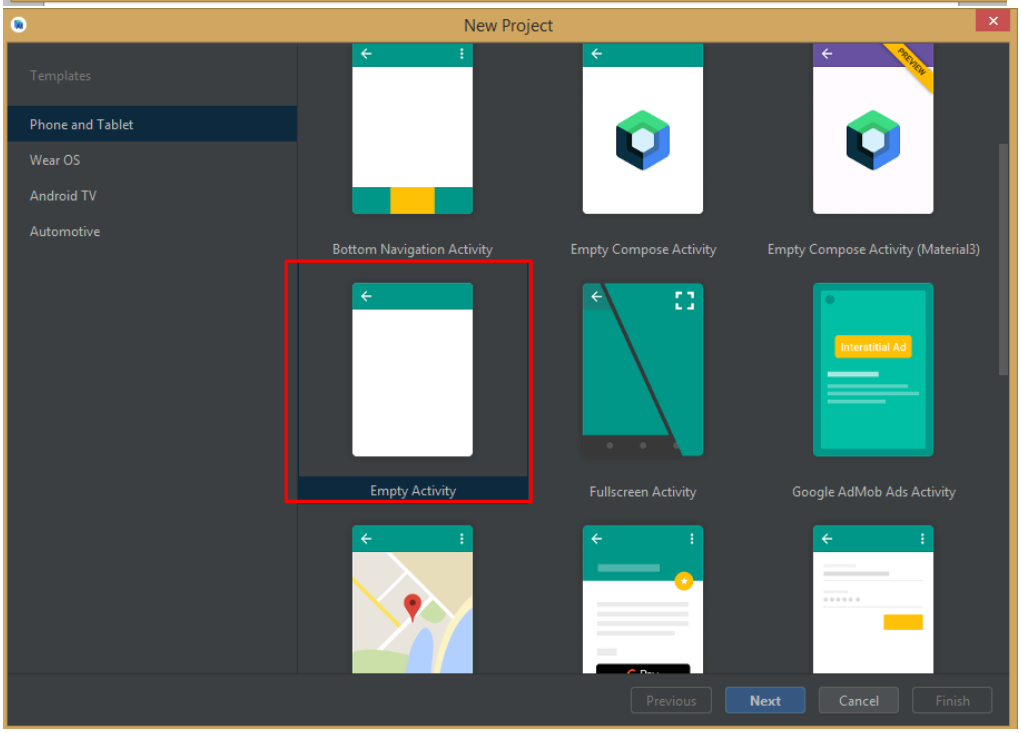
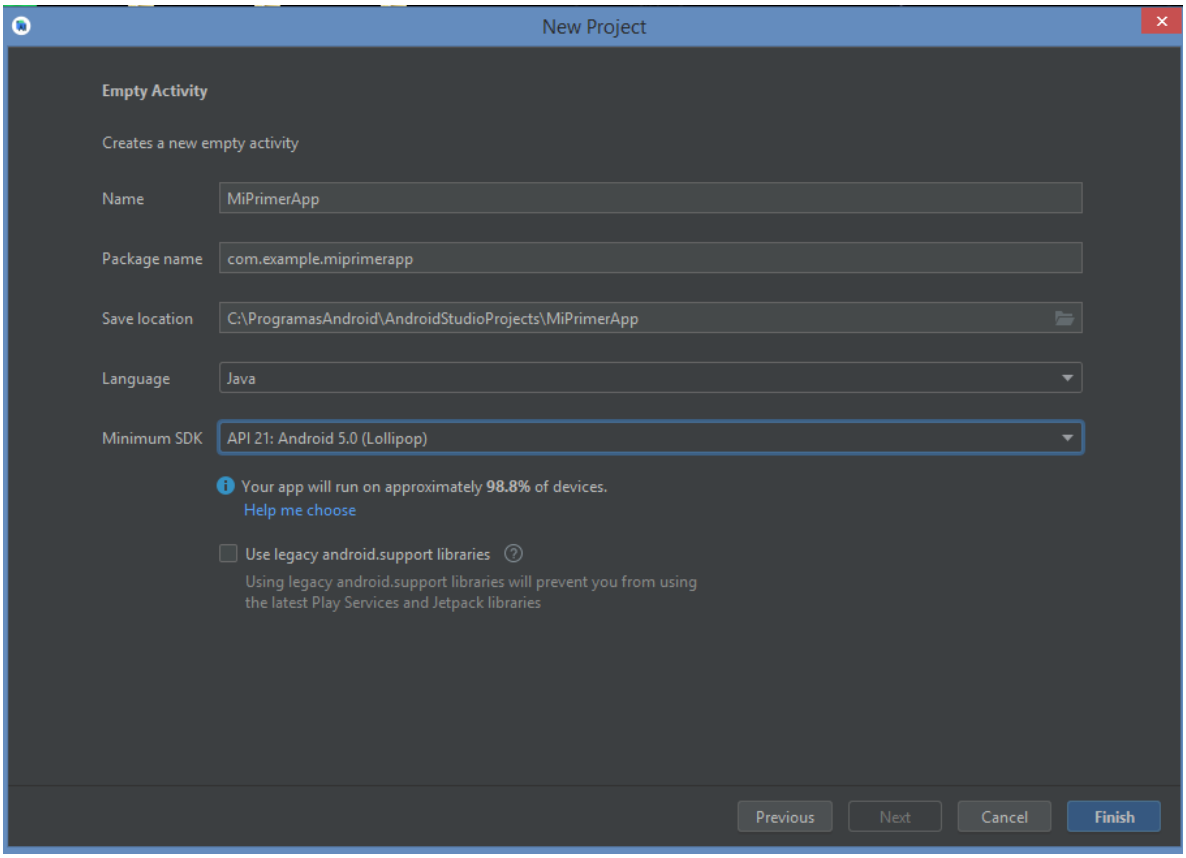
MANUAL BASICO ANDROID STUDIO

Índice

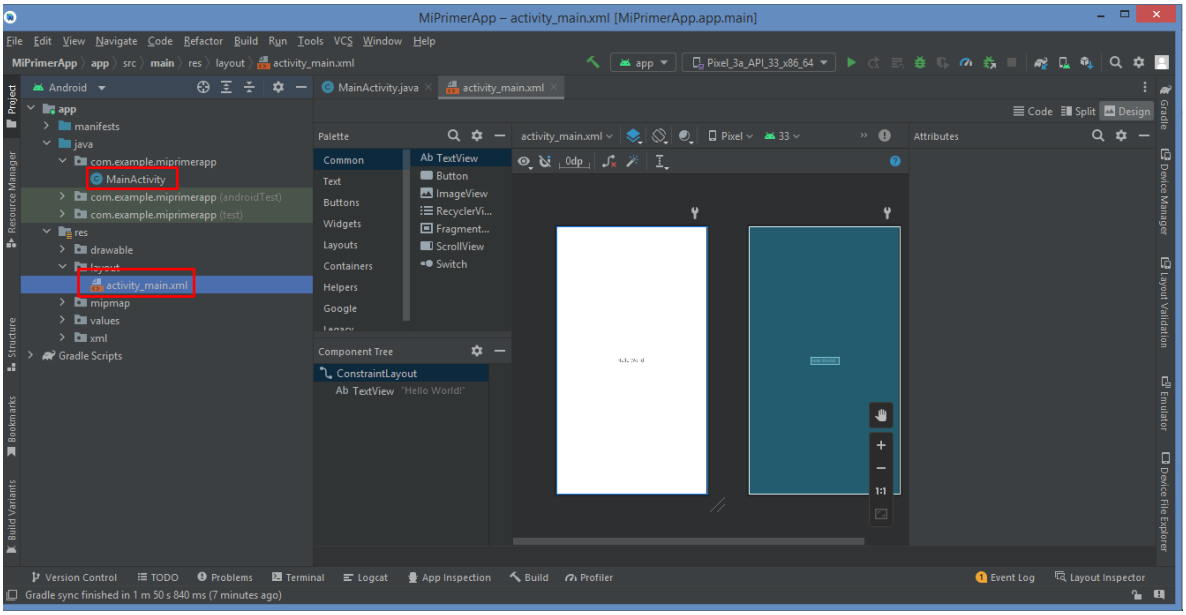
1) Mi primer programa





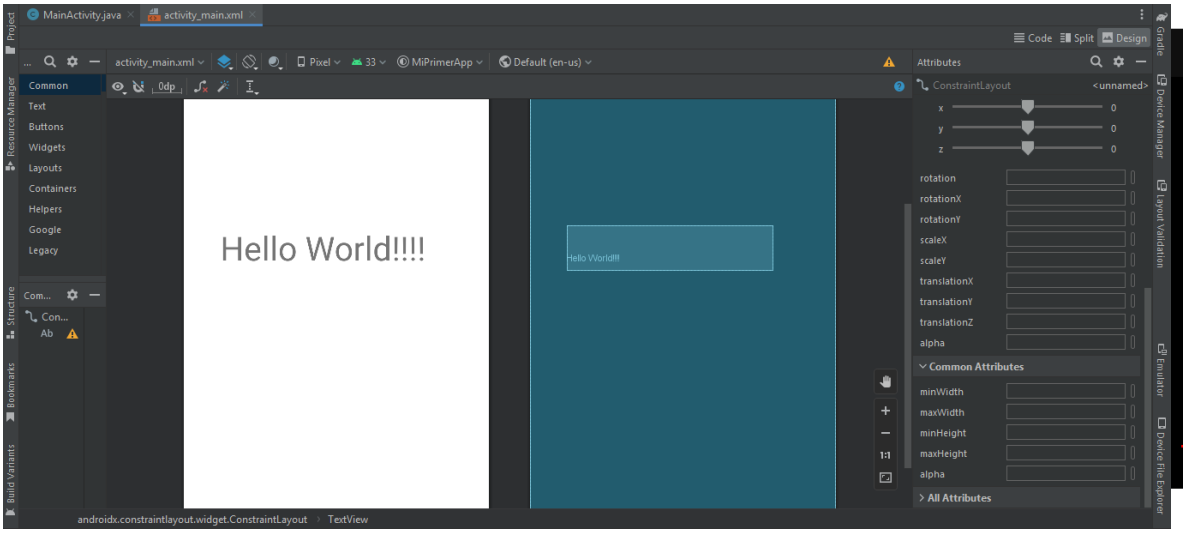


Al abrir el Proyecto nos abrirá un archivo java que será nuestra parte lógica y un archivo xml que será la parte visual, los archivos podemos abrirlos desde la barra de navegación de la izquierda



a) La herramienta ConstraintLayout

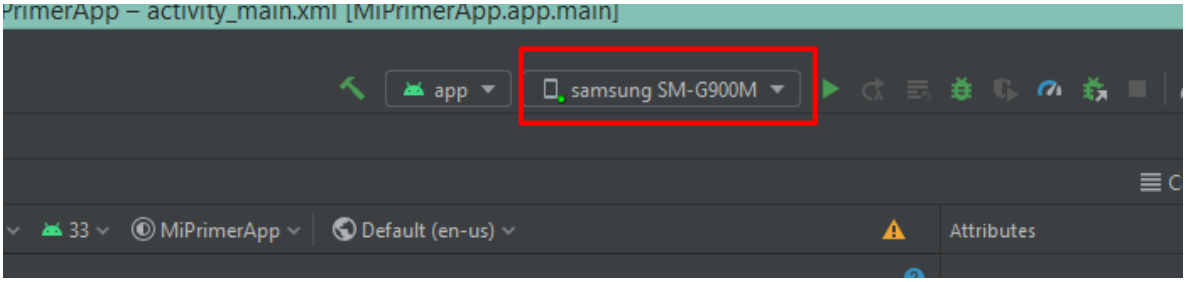
El archive xml nos permite trabajar la interfaz que vera el usuario y nos permite intercalar en el diseño y el código



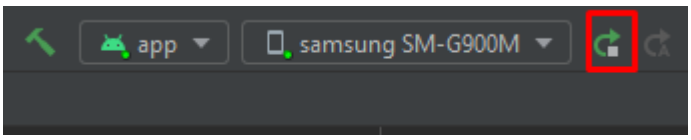
La ventaja de esta herramienta es que no necesitamos dominar html para poder hacer el diseño de la aplicación ya que la misma ventana de diseño para haciendo el código por nosotros, solo de la parte de diseño la parte lógica la haremos nosotros

2) Conectar dispositivo físico

Para conectar un dispositivo para facilitar la prueba de las aplicaciones debemos activar las opciones de desarrollador o programador y activar la depuración por usb, en automático Android studio nos detectara el dispositivo y una vez terminado el programa solo bastara con presionar el símbolo de play verde que esta aun lado del nombre de nuestro dispositivo, Android studio se encargara de instalar el programa en nuestro dispositivo.

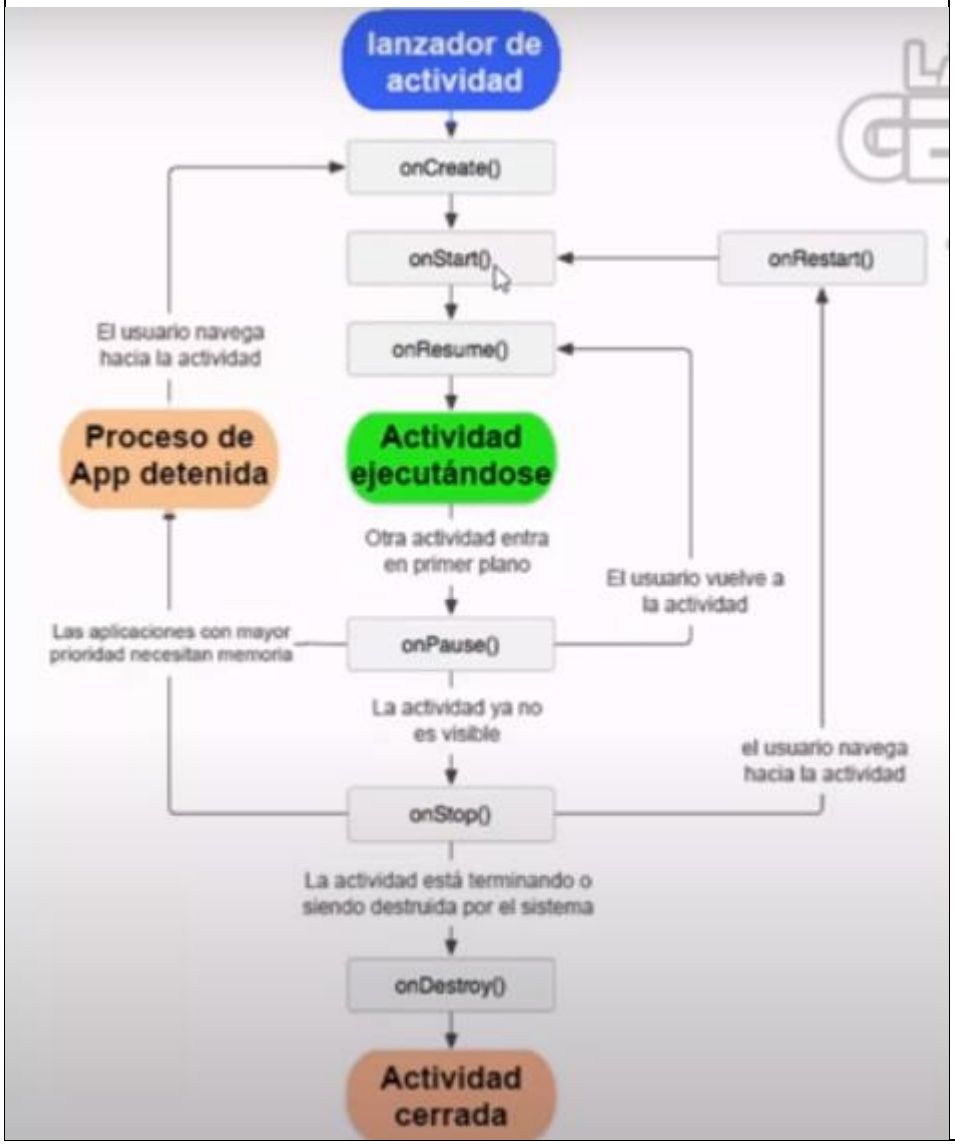


Si el programa ya está instalado en el celular, aparecerá en lugar del símbolo de play color verde, saldrá una flecha curva.



También se recomienda en configuración del dispositivo tener activada la casilla de orígenes desconocidos y desactivar play protect

3) Ciclo de vida de un activity o actividad



onCreate : Se ejecuta para dar inicio a la actividad, si no se coloca no pasaremos al siguiente punto

onStar: Para iniciar el activity

onResume: para visualizar el activity

onPause: pausar, es decir pasar de primer a segundo plano

onStop: ocultar activity

onDestroy: cierra o destruye la activity

onRestart: volver a activar el activity

4) Ejemplo de ciclo de vida de una activity

Volviendo al ejercicio de hola mundo, en al archivo java podemos observar que ya está el método onCreate

```java
package com.example.miprimerapp;

import ...

public class MainActivity extends AppCompatActivity {

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

Ahora copia el siguiente código y pégalo en el editor como se muestra :

```
@Override
  protected void onStart() {
    super.onStart();
    Toast.makeText(this, "OnStart", Toast.LENGTH_SHORT).show();
    // La actividad está a punto de hacerse visible.
  }
  @Override
  protected void onResume() {
    super.onResume();
    Toast.makeText(this, "OnResume", Toast.LENGTH_SHORT).show();
    // La actividad se ha vuelto visible (ahora se "reanuda").
  }
  @Override
  protected void onPause() {
    super.onPause();
    Toast.makeText(this, "OnPause", Toast.LENGTH_SHORT).show();
    // Enfocarse en otra actividad  (esta actividad está a punto de ser "detenida").
  }
  @Override
  protected void onStop() {
    super.onStop();
    Toast.makeText(this, "OnStop", Toast.LENGTH_SHORT).show();
    // La actividad ya no es visible (ahora está "detenida")
  }
  @Override
  protected void onDestroy() {
    super.onDestroy();
    Toast.makeText(this, "OnDestroy", Toast.LENGTH_SHORT).show();
    // La actividad está a punto de ser destruida.
  }
```

```java
package com.example.miprimerapp;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    @Override
    protected void onStart() {
        super.onStart();
        Toast.makeText(this, "OnStart", Toast.LENGTH_SHORT).show();
        // La actividad está a punto de hacerse visible.
    }
    @Override
    protected void onResume() {
        super.onResume();
        Toast.makeText(this, "OnResume", Toast.LENGTH_SHORT).show();
        // La actividad se ha vuelto visible (ahora se "reanuda").
    }
    @Override
    protected void onPause() {
        super.onPause();
        Toast.makeText(this, "OnPause", Toast.LENGTH_SHORT).show();
        // Enfocarse en otra actividad  (esta actividad está a punto de ser "detenida").
    }
    @Override
    protected void onStop() {
        super.onStop();
        Toast.makeText(this, "OnStop", Toast.LENGTH_SHORT).show();
        // La actividad ya no es visible (ahora está "detenida")
    }
    @Override
    protected void onDestroy() {
        super.onDestroy();
        Toast.makeText(this, "OnDestroy", Toast.LENGTH_SHORT).show();
        // La actividad está a punto de ser destruida.
    }
}
```

Como puedes observar ahora tenemos cada uno de los métodos que describimos en el punto anterior.

Ahora necesitamos usar Toast aún no veremos que es, solo hay que saber que lo necesitamos y para usarlo necesitamos importar una librería, si te posicionas en alguna palabra en rojo, en este caso Toast puedes presionar ALT+ENTER

```
package com.example.miprimerapp;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    @Override
    protected void onStart() {
        super.onStart();
        Toast.makeText( context: this,  text: "OnStart", Toast.LENGTH_SHORT).show();
        // La actividad está a punto de hacerse visible.
    }
    @Override
```

La librería se agregó y no nos marca el error

Copia y pega las siguientes líneas de texto, pégalas en el método onCreate antes de la llave de cierre.

```
Toast.makeText(this, "OnCreate", Toast.LENGTH_SHORT).show();
// La actividad está creada.
```

Asi nos quedaria:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Toast.makeText( context: this,  text: "OnCreate", Toast.LENGTH_SHORT).show();
    // La actividad está creada.
}
```

Observa que en automático el editor nos muestra cual sería el texto a mostrar, esto solo para que te familiarices.

Ejecuta el programa en tu Android deberá pasar los siguiente, al iniciar saldrá la leyenda onCreate, inmediatamente salda el enunciado onStart y onResume. Si presionas la tecla home o te vas a la pantalla principal del dispositivo saldrá la leyenda onPause, onStop y onDestroy

5) Debuggeo (revisar errores)

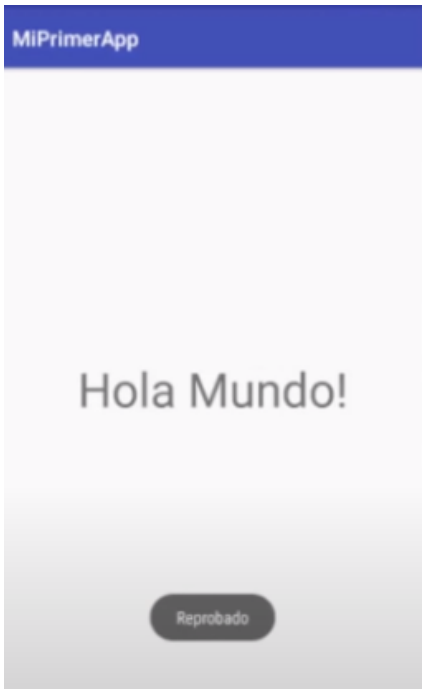
El debugeo o depurador es utilizado para encontrar los bugs o errores, para ello necesitamos un break point, nos iremos a la parte lógica de nuestro programa y borraremos todos los método menos el onCreate, y borraremos las ultimas 2 lineas que agregamos, nos quedaría asi

```
package com.example.miprimerapp;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

    }
}
```

Ahora donde se muestra el puntero empezaremos a programar, haremos un programa para calcular el promedio de 3 calificaciones, y con un if imprimir si aprobó o reprobo

```
package com.example.miprimerapp;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
      int mate=5, español=5, fisica= 5;
      int promedio= (mate+español+fisica)/3;
      if (promedio>=6){
          Toast.makeText(this,"aprobado",Toast.LENGTH_SHORT).show();
      }else if(promedio<6){
          Toast.makeText(this,"reprobado",Toast.LENGTH_SHORT).show();
      }
    }
}
```

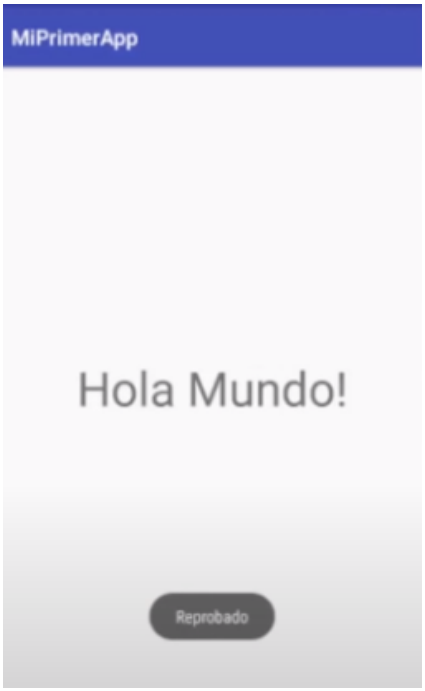
nos saldra este mensaje



Ahora vamos a crea un error para poder revisar el debuggeo, cambiaremos la condición if

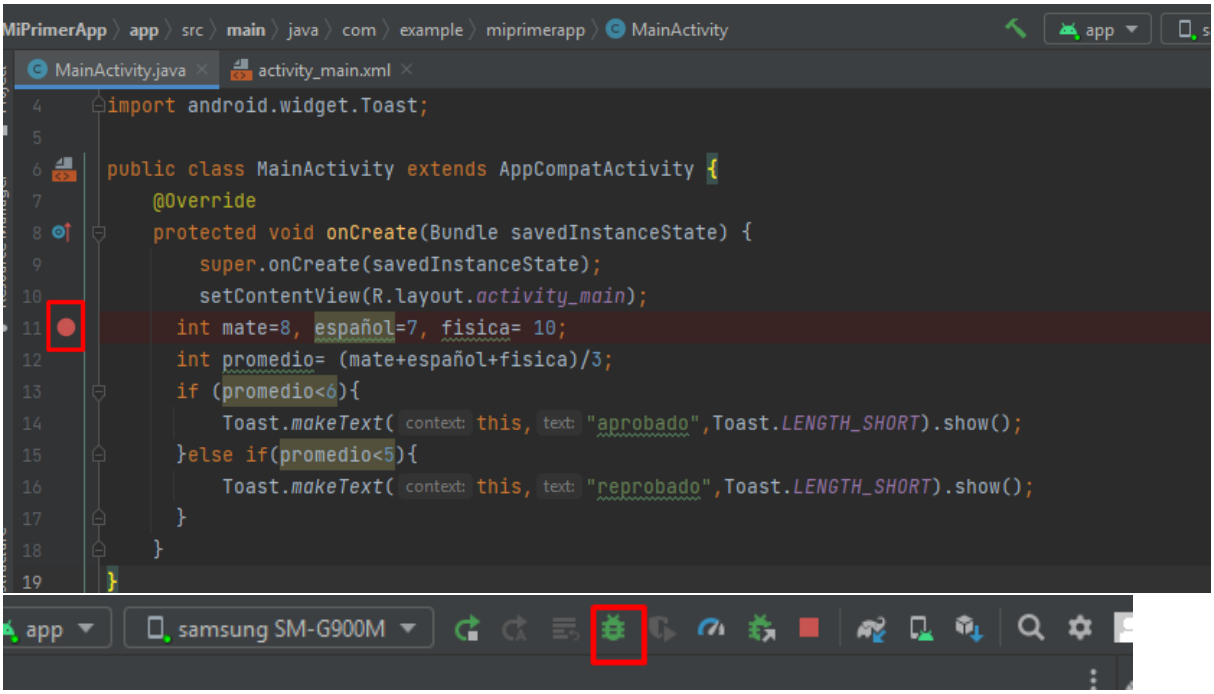
```
if (promedio<6){
    Toast.makeText(this,"aprobado",Toast.LENGTH_SHORT).show();
}else if(promedio<5){
    Toast.makeText(this,"reprobado",Toast.LENGTH_SHORT).show();
}
```

Otra vez sale el mensaje de aprobado

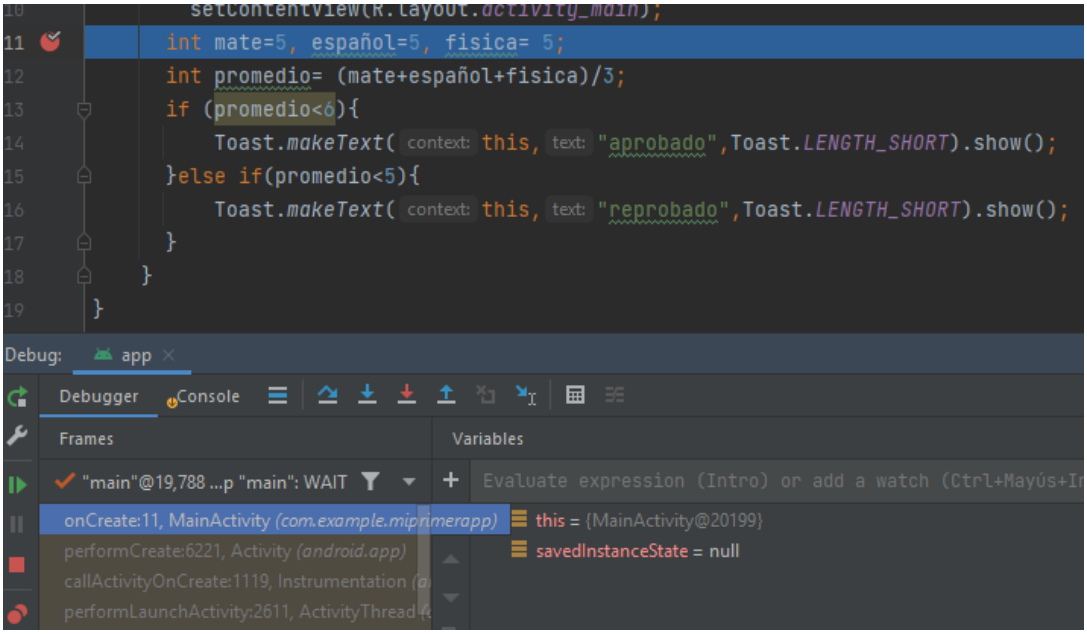


Ahora para iniciar el debugeo necesitamos el break point, para colcoarlo solo debes de dar clic a la línea de código donde quieres colocarlo

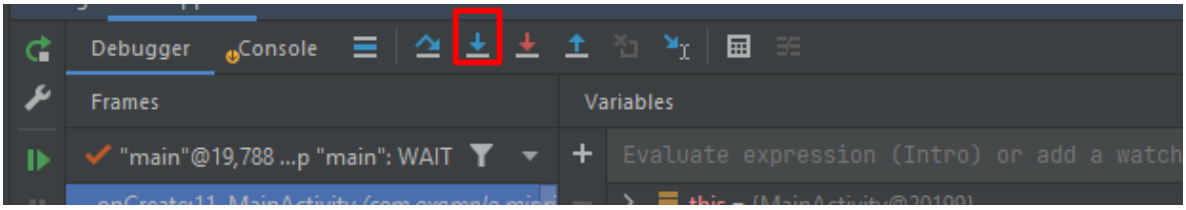
Colocamos el breakpoin y presionamos el símbolo de debugeo



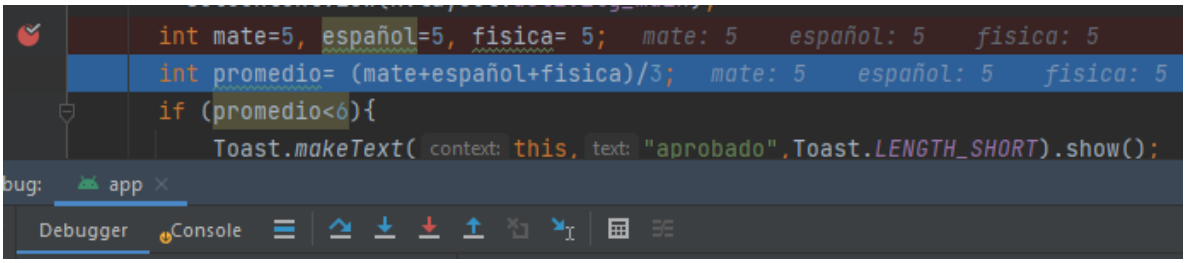
El programa nos abrirá esta nueva ventana de control



Con estas flechas nosotros controlaremos el proceso, con esa iremos indicando que pasemos a la siguiente línea de código (también podemos usar F7)

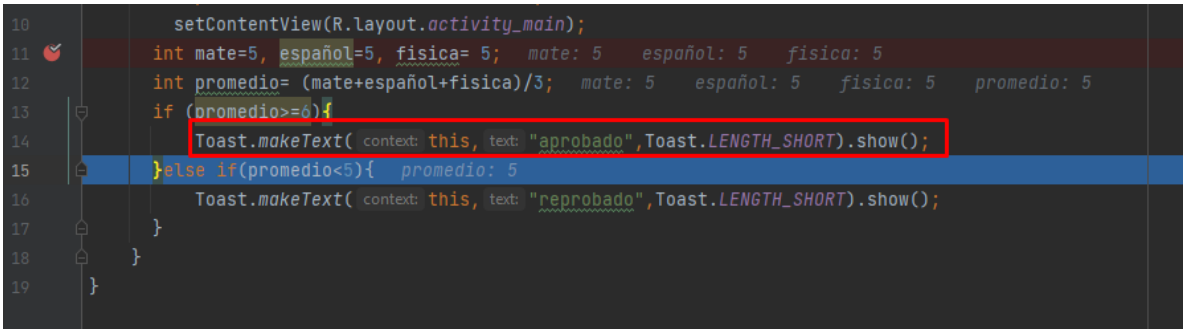


Ahora presionamos y observamos que el editor nos dice que se declaró la variable mate, español y física y cuánto vale cada una



Damos clic para la siguiente línea, ahora aquí nos dice el valor del promedio con esto podemos observar que se ejecutara esta línea porque cumple la condición

Cambiemos el código a como estaba antes



Ahora al debuggear podemos observar que el programa se salta esa línea pues no cumple la condición

6) Mensaje emergente con la clase Toast

Como ya observamos en los ejercicios pasados los toast son mensajes emergentes que damos al usuario, aparece en la activity sin afectar el funcionamiento, pueden ser textos, imágenes o ambos, para usarlo necesitamos la librería

```
Import Android.widget.Toast;
```

Aunque el editor lo coloca por defecto cuando colocamos un Toast, la estructura de un toast es la siguiente

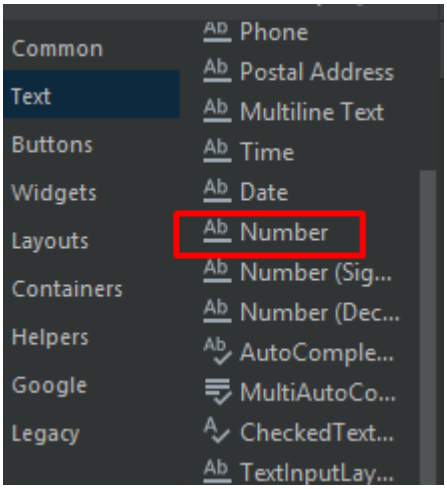
```
Toast.makeText(Contexto,"Texto", duración).show();
```

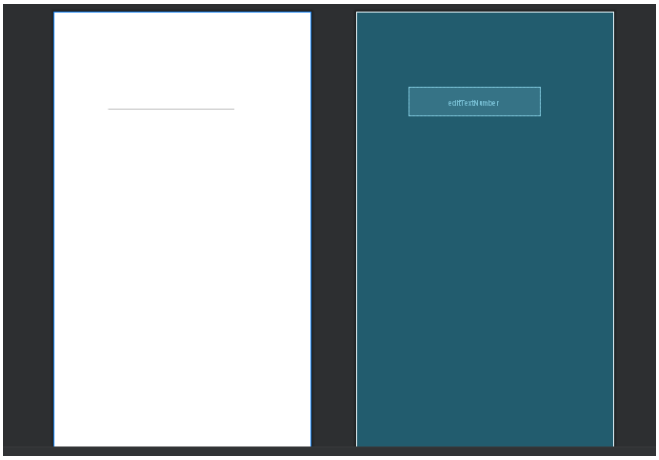
En la parte de contexto colocamos la palabra reservada This, para indicar que se ejecutara en este activity, en duración pondremos lo siguiente:

Toast.LENGTH_SHORT para medio segundo
Toast.LENGTH_LONG para un segundo completo

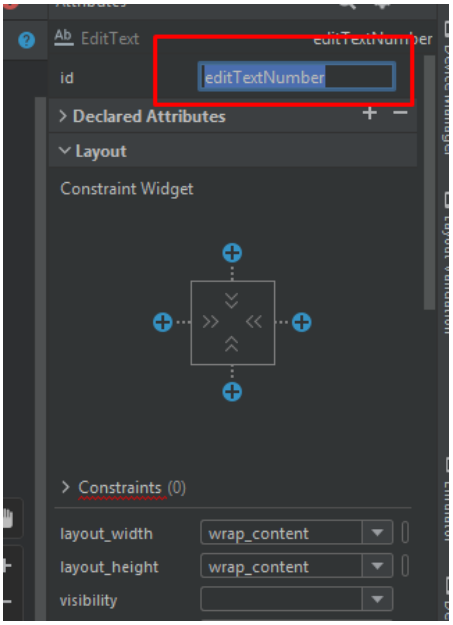
7) Realizando una calculadora (parte grafica)

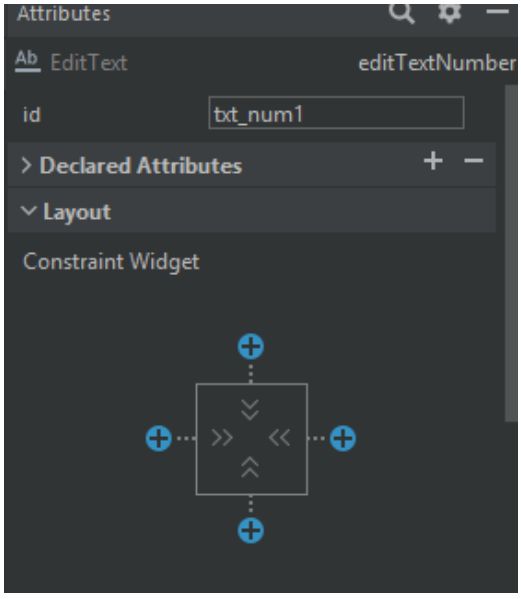
Crea otra activiy vacía, lo primero que vamos a diseñar será la parte grafica
Iremos a la parte gráfica y seleccionaremos el componente y arrastraremos a nuestra pantalla de diseño (la pantalla blanca)



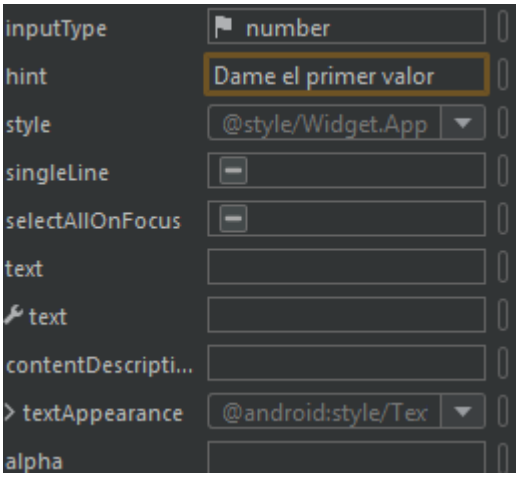


Ahora lo primero que debes hace siempre que pones un componente es darle un ID único, seleccionamos el componente y nos vamos a la lista de atributos que esta de lado derecho

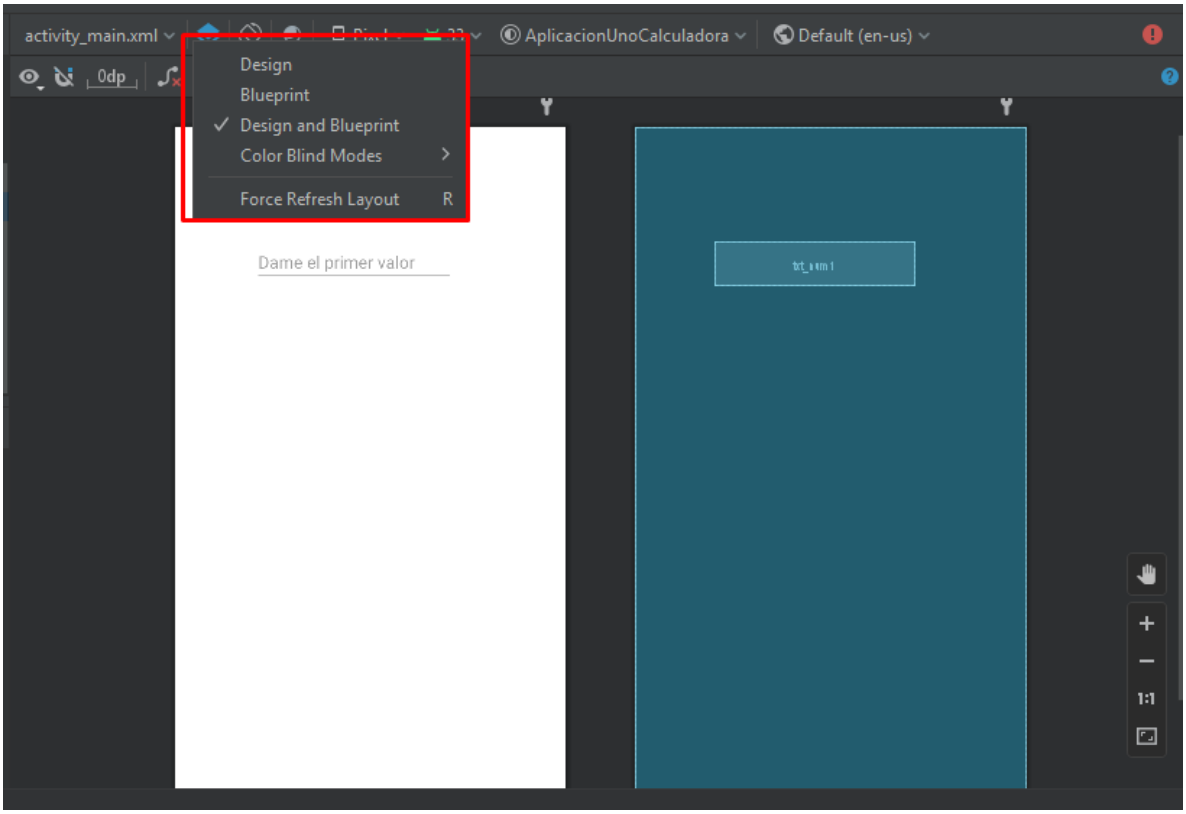




En este caso le pusimos el nombre de txt_num1 para decir que es la entrada de texto que guardara al número 1, ahora colocaremos algún mensaje que le aparezca al usuario para ello buscaremos hint (para que salga transparente el texto y que desaparezca al momento de escribir) y pondremos dame el primer valor

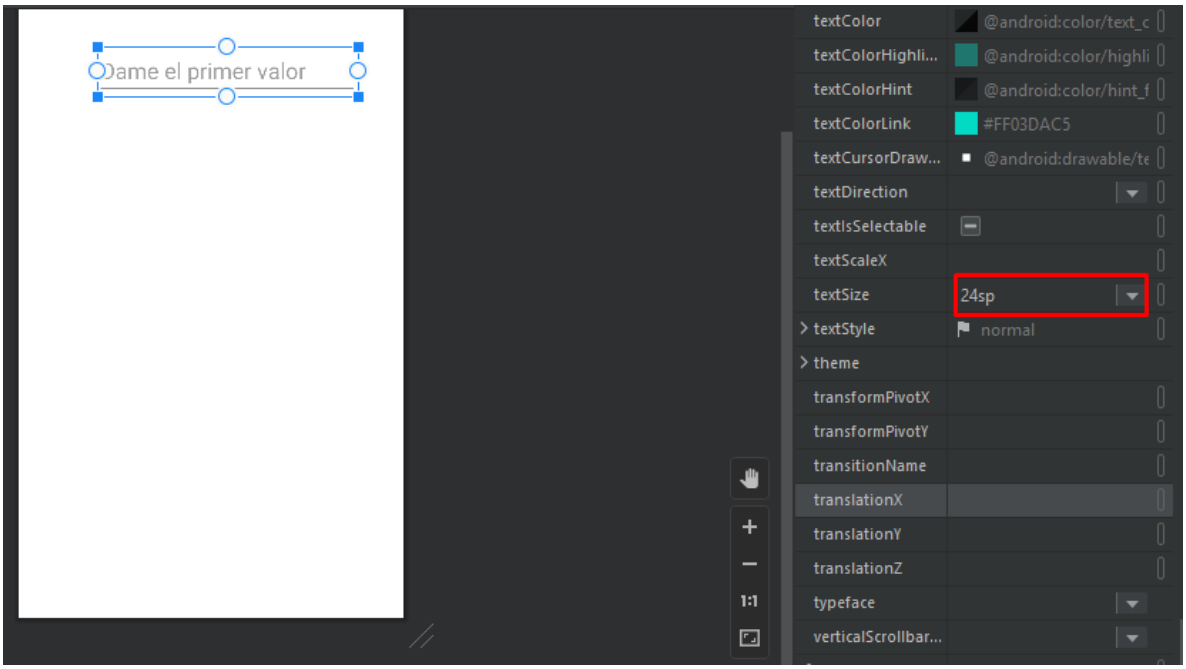


Ahora como tip podemos alternar las vistas que queremos, para no saturarnos dándole clic a ese boton

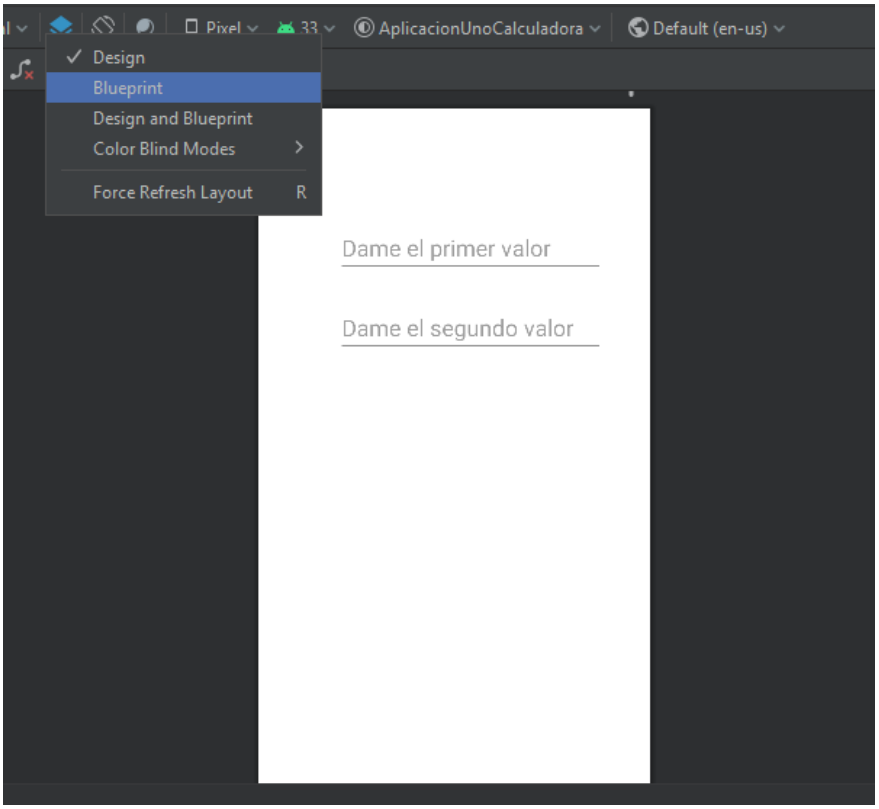


Haremos más grande las letra, así que vamos a atributos otra vez y buscamos la opción de

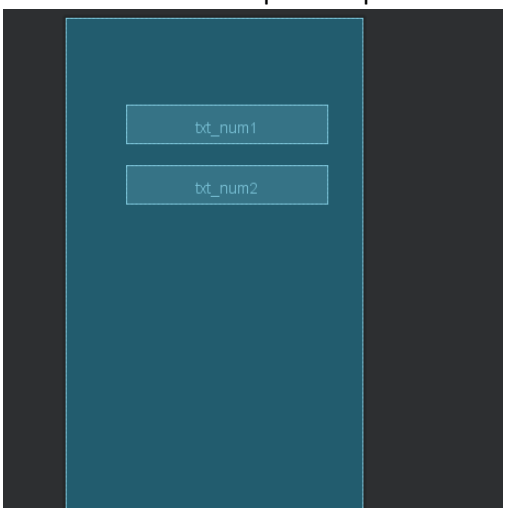
Y desplegamos la lista, ahora buscamos text size (tamaño de texto) y jugamos con los tamaños que vienen hasta que encontremos uno de nuestro agrado



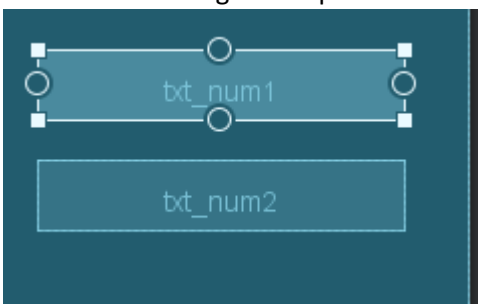
Coloquemos el segundo text para recibir el segundo valor y nos vamos a la vista blueprint

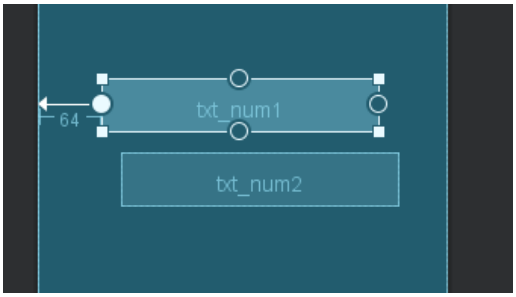


Podemos considerar que esta pantalla es como la radiografía de nuestro diseño

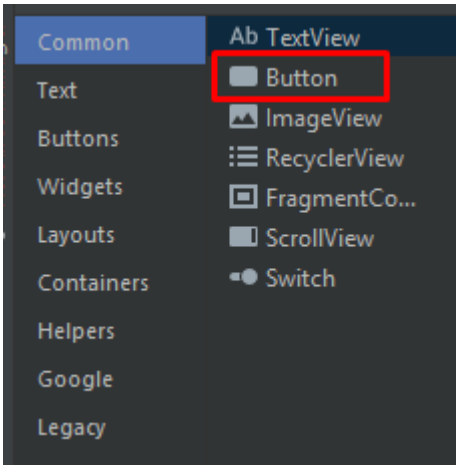


Si hacemos clic en algún componente nos saldrá de la siguiente forma

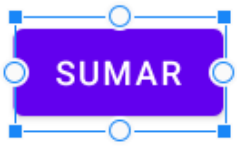


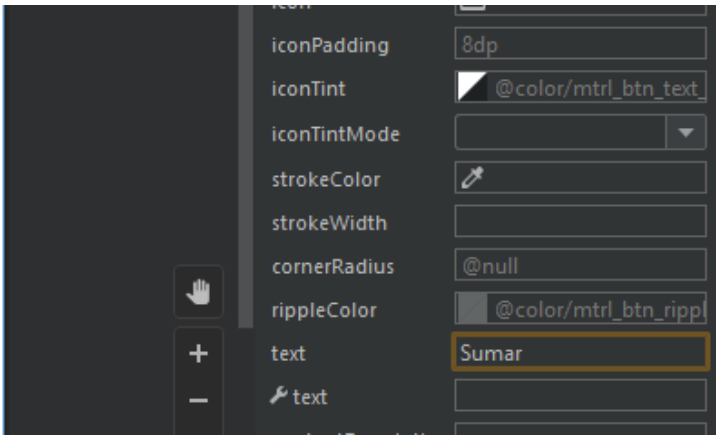


Estos puntos nos permiten fijar el componente ya sea algún borde o algún otro componente o bien dar las medidas de separación para que nuestra app mantenga una vista agradable, se recomienda jugar un poco con esto para familiarizarse y que el primer text este enlazado a uno de los lado y a la parte superior de la pantalla, ahora regresamos a la vista diseño y agregaremos un botón

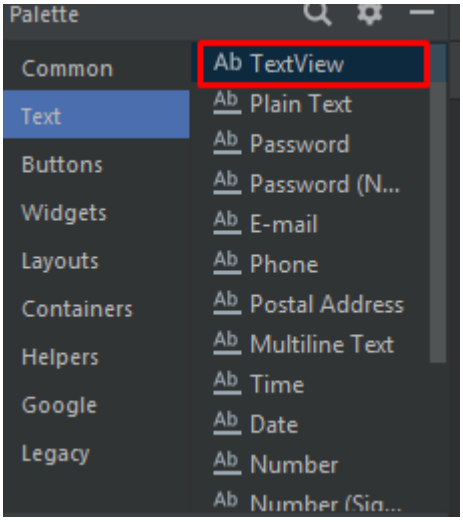


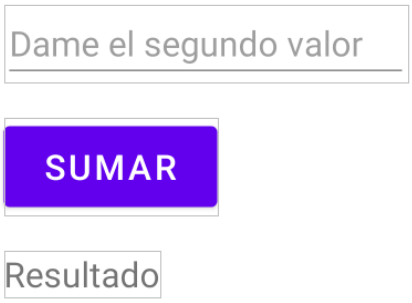
Ahora al botón no le pondremos el nombre con la opción hint ya que no queremos que desaparezca ni que se vea transparente buscaremos en atributos la parte donde dice text





Y asigna un ID al botón en este caso botón_sumar, ahora necesitamos donde se muestre el resultado, y que no se pueda editar pondremos un Text view, le pondremos un Id en este caso txt_resultado, en la parte donde dice Text escribiremos Resultado y cambiemos su tamaño





Finalmente recuerda volver a la pantalla blueprint y ajustar el botón y el textview para evitar que se mueva

8) Realizando una calculadora (parte lógica)

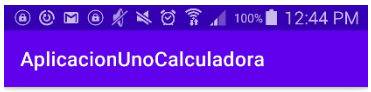
Codigo

```
package com.example.aplicacionunocalculadora;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
  private EditText et1, et2; //para recibir datos
// private para que solo se use en esta clase, EditText para que podamos
modificar, inmediatamente definimos nuestra variables
// no se necesario poner aun el id de nuestros valores
  private TextView tv1; //para mostrar resultado
// private para que solo se use en esta clase, TextView para que podamos
modificar, inmediatamente definimos nuestra variables
// no se necesario poner aun el id de nuestros valores

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
//EMPEZAMOS PARTE LOGICA O DE ASIGNACION
// escribimos la variable y asignamos
// hacemos un casting o conversion ponemos entonces (EditText)
// comando findViewById() que significa encuentra en la vista por el ID
// entrte parentesis R.id.IDdeNuestroObjeto
// R es la clase si vemos mas arriba ya la usamos
        et1= (EditText) findViewById(R.id.txt_num1);
        et2= (EditText) findViewById(R.id.txt_num2);
 // en este caso cambia el casting
        tv1= (TextView) findViewById(R.id.txt_resultado);
    }
//PARTE FUNCIONAL
// creamos un metodo publico y void porque no regresara un valor de tipo
especifico
// entre parentesis dcelaramos una variable tipo View y le ponemos un
nombre en este caso view1
public void Sumar(View view1){
//Definimos las varibales tipo texto y asignamos a nuestras variables
donde guardamos
// lo recibido desde pantalla et1 y et2
//con getText(). le decimos que recupere el texto y convertimos
(parciamos) a
// tipo string con .toString()
//parcial = analizar gramaticalmente
   String valor1 =et1.getText().toString();
   String valor2 =et2.getText().toString();
// declaramos variables tipo entero
// Integer.parseInt() para indicar la conversion a valor tipo integer o
entero,
// entre parentesis el valor a parciar
   int num1 = Integer.parseInt(valor1);
   int num2 = Integer.parseInt(valor2);
```

```
// realizamos operacion
   int suma = num1+num2;
// ahora guardamos el resultado en una variable string
// definimos que sera tipo String hacemos uso del metodo .valueOf(),
entre parentesis el valor
String result = String.valueOf(suma);
//asignamos a la variable tv1 con aydua del metodo setText() (asigna
texto)
tv1.setText(result);
    }

}
```

Ahora vamos a la parte gráfica y seleccionamos el botón sumar y en la parte de atributos, buscamos la opción onClick y seleccionamos nuestro método sumar

Resultado



10

5

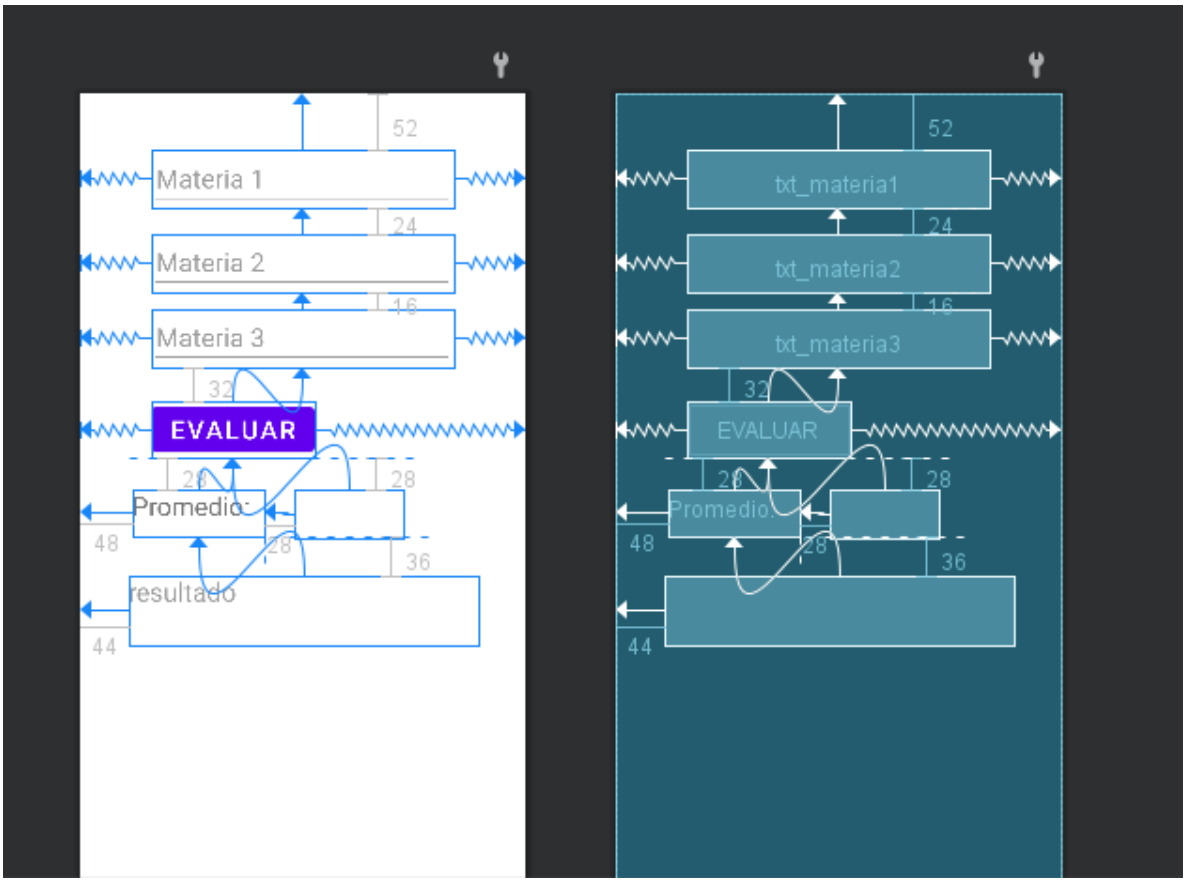
SUMAR

15

9) Ejercicio practico programa de evaluación (parte grafica)

Realiza una app donde puedas introducir la calificación de 3 materias y evaluar si el alumno aprobó o reprobó
Colocaremos 3 text tipo numérico para materia 1 ,2 y 3, 1 boton que se llama evaluar y 3 text uno para tener la palabra promedio, otro para mostrar el resultado del promedio y otro será textview para mostrar el resultado

Tipo	id	Hint	Text
Text numérico	Txt_materia1	Materia1	-
Text numérico	Txt_materia2	Materia2	-
Text numérico	Txt_materia3	Materia3	-
Text View	prom	-	Promedio:
Text View	Res_pom	-	-
Text View	Resultado	-	-

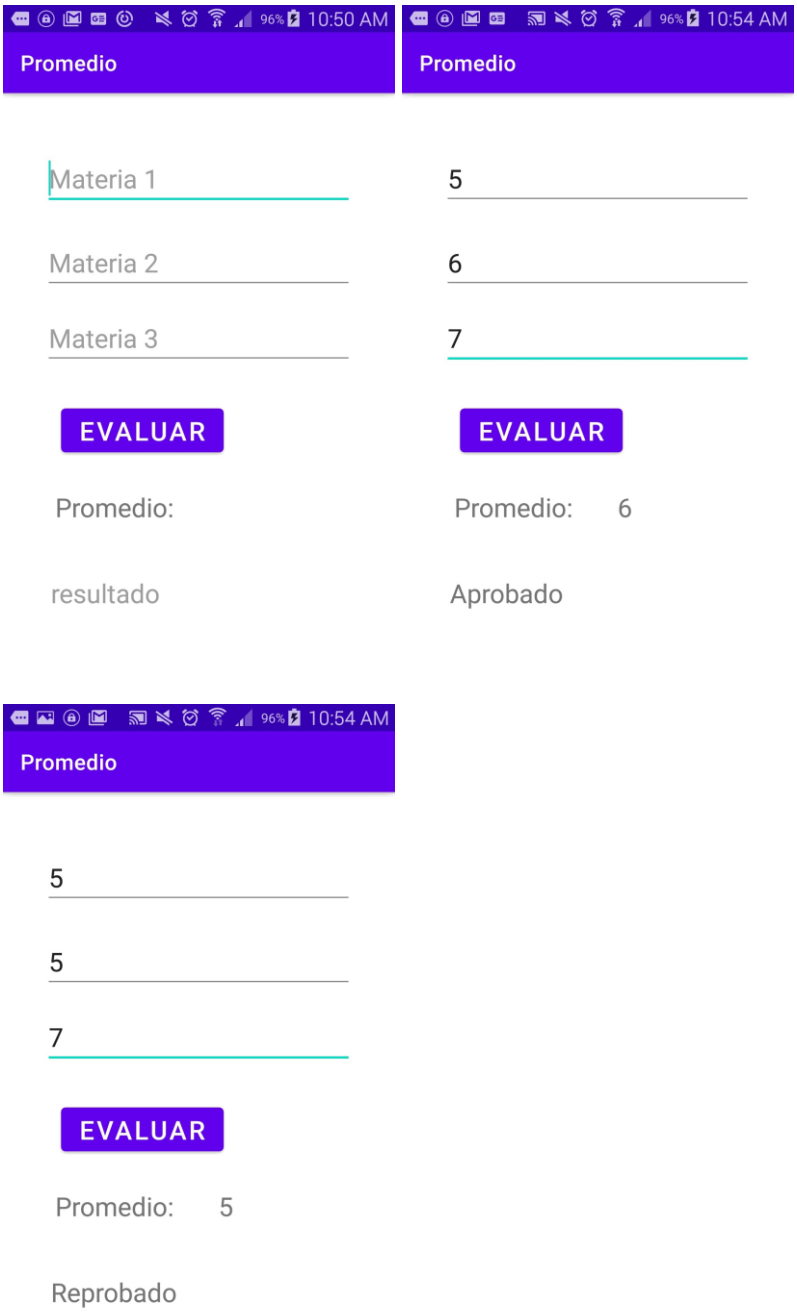


10) Ejercicio practico programa de evaluación (parte lógica)

Codigo

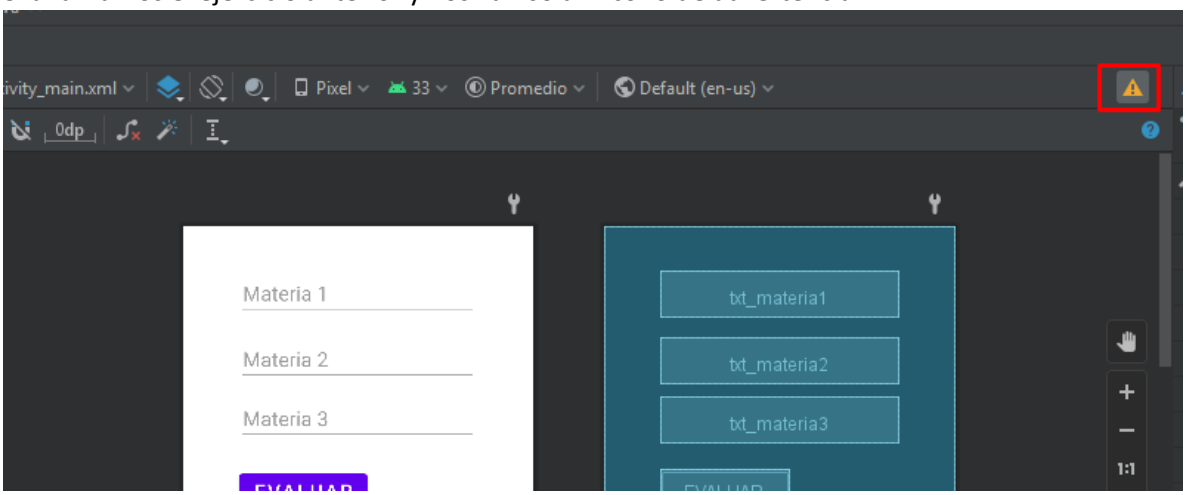
```
package com.example.promedio;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
   //definimos variables privadas
    private EditText et1 , et2 , et3;
    private TextView tv1, tv2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //asignamos los elementos de panatlla a cada variable
        et1 =(EditText) findViewById(R.id. txt_materia1);
        et2 =(EditText) findViewById(R.id. txt_materia2);
        et3 =(EditText) findViewById(R.id. txt_materia3);
        tv1= (TextView) findViewById(R.id.resultado);
        tv2= (TextView) findViewById(R.id.res_prom);
    }

    public void evaluar(View view1){
//declaramos variables tipo String y convertimos tipo texto a string
        String valor1 = et1.getText().toString();
        String valor2 = et2.getText().toString();
        String valor3 = et3.getText().toString();
//declaramos variable tipo integer y convertimos de string a integer
        int num1= Integer.parseInt(valor1);
        int num2= Integer.parseInt(valor2);
        int num3= Integer.parseInt(valor3);
// declaramos y realizamos promedio
        int promedio = (num1+num2+num3)/3;
// declaramos variables para guardar resultados
        String condicion="", result="";
//condicion para aprobado y reprobado
        if(promedio>=6){
            condicion = "Aprobado";
        }if(promedio<6){
            condicion = "Reprobado";
        }
//asignamos a text view y convertimos de string a texto
        tv1.setText(condicion);
//convertimos de integer a string
        result =  String.valueOf(promedio);
// asignamos a text view y convertimos de string a texto
        tv2.setText(result);
    }
}
```

Resultado

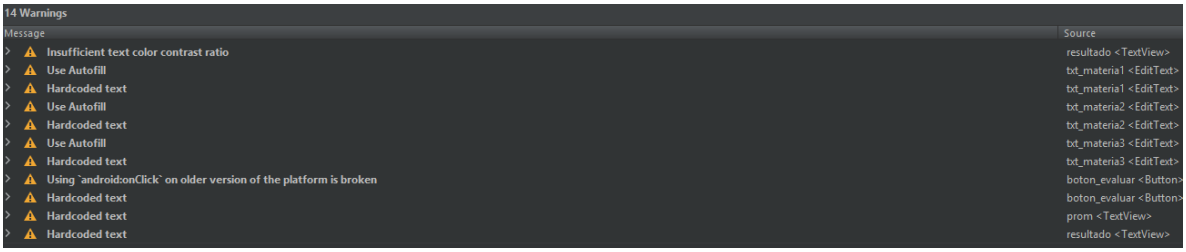

10:50 AM
Promedio
Materia 1
Materia 2
Materia 3
EVALUAR
Promedio:
resultado
10:54 AM
Promedio
5
6
7
EVALUAR
Promedio: 6
Aprobado
10:54 AM
Promedio
5
5
7
EVALUAR
Promedio: 5
Reprobado

11) Hardcode string shoul use string resource

Si analizamos el ejercicio anterior y nos vamso al incono de advertencia

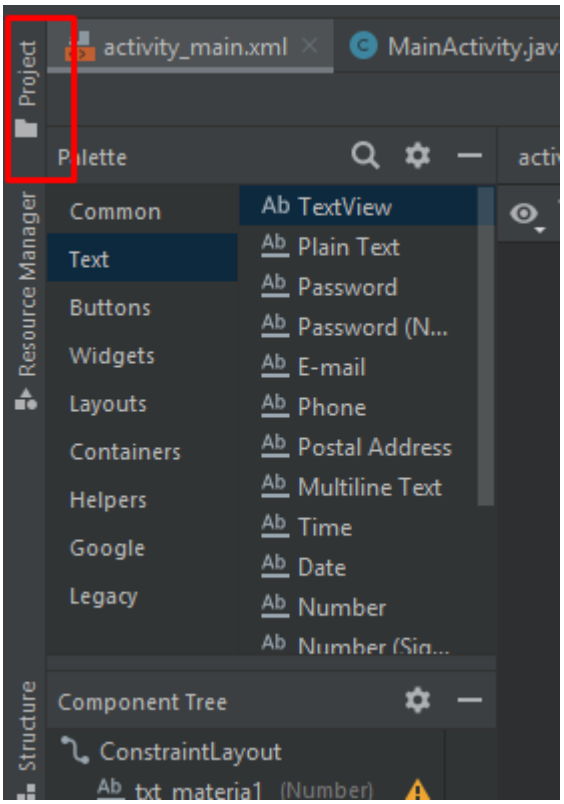


Nos sale lo siguiente

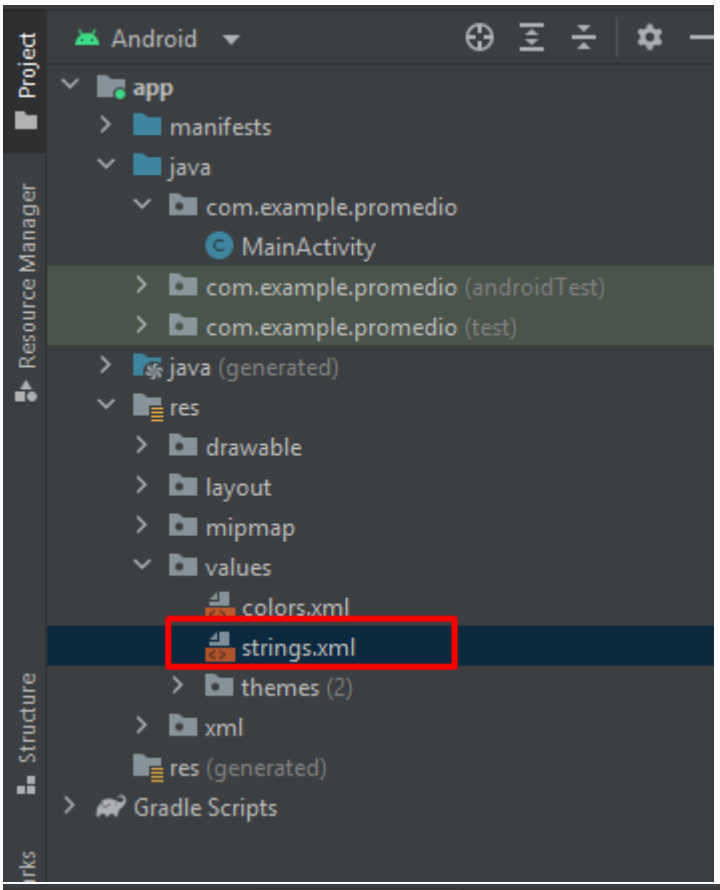


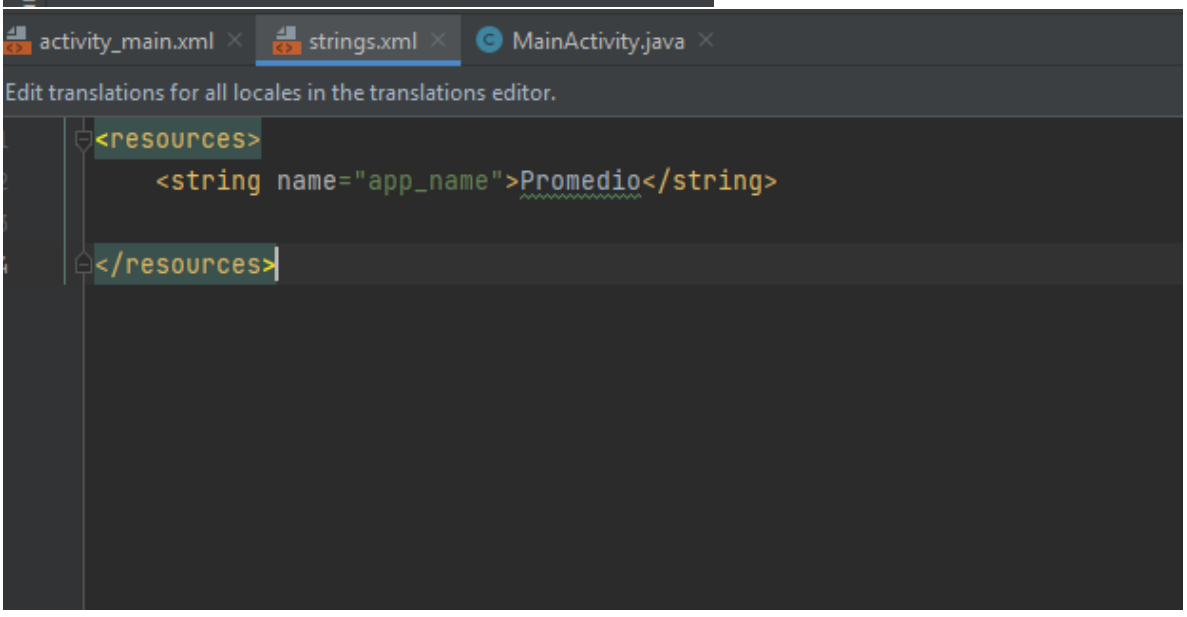
Estos no son errores son advertencias, para quitar esas advertencias haremos uso del archivo string.

Vamos al Android studio a Project



Y abrimos el archivo string





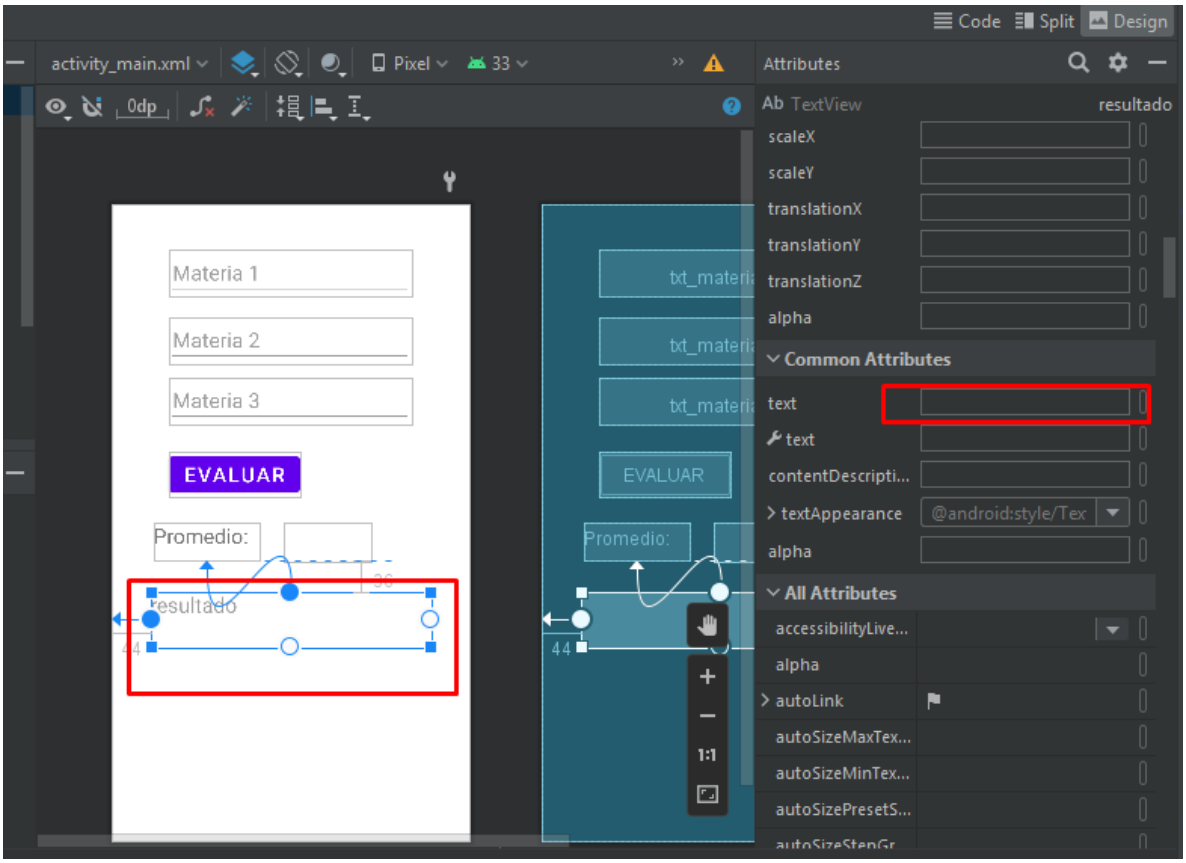
Aquí empezaremos a trabajar y colocar código en html, pero es muy sencillo, solo seguimos la línea de código que ya esta

De manera de ejemplo

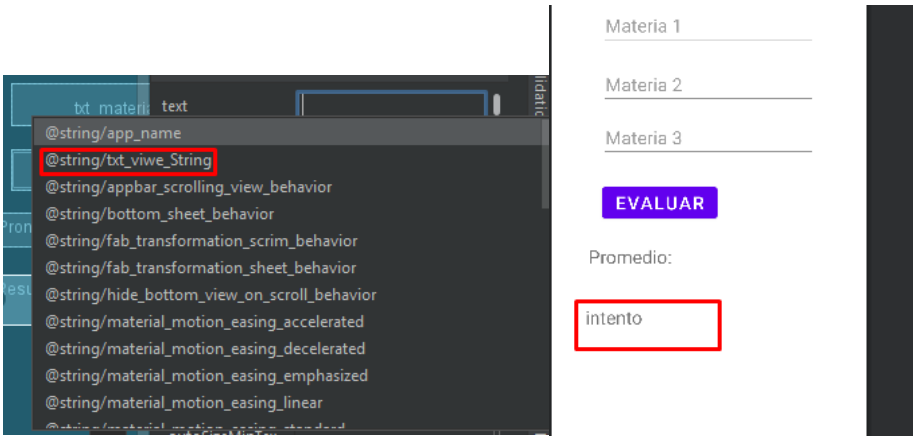
<string name="txt_viwe_String">Intento</string>
<String name="podemosPonerElNombreDeVariableQueQueramos">NombreQuequeramos</strin>

```
<resources>
    <string name="app_name">Promedio</string>

    <string name="txt_viwe_String">intento</string>
</resources>
```

Es comun que Android studio nos ayude a completar el código, una vez declarado vamos al archivo xml Seleccionamos el componente que vamos a definir, en este caso el Textviem de resultado y en atributos buscamos donde diga text

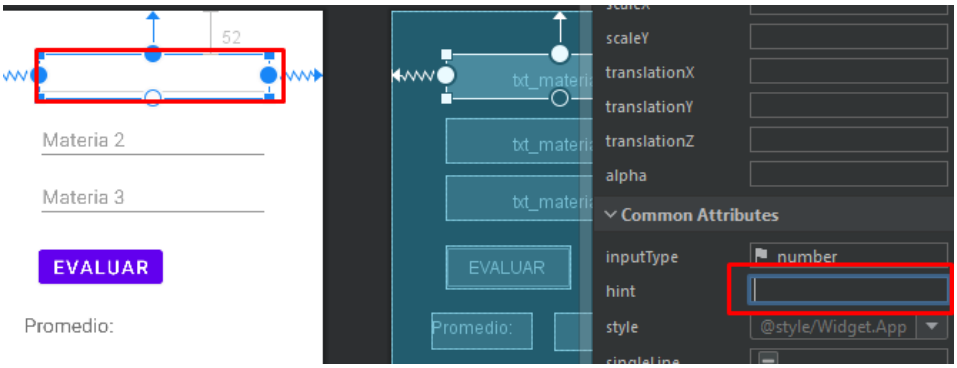


Ahora presionamos la combinación CTRL+BarraDeSpacio y nos saldrá la variable que acabamos de definir y la seleccionamos, y podemos observar que en el diseño cambio el nombre del TextView ahora dice intento

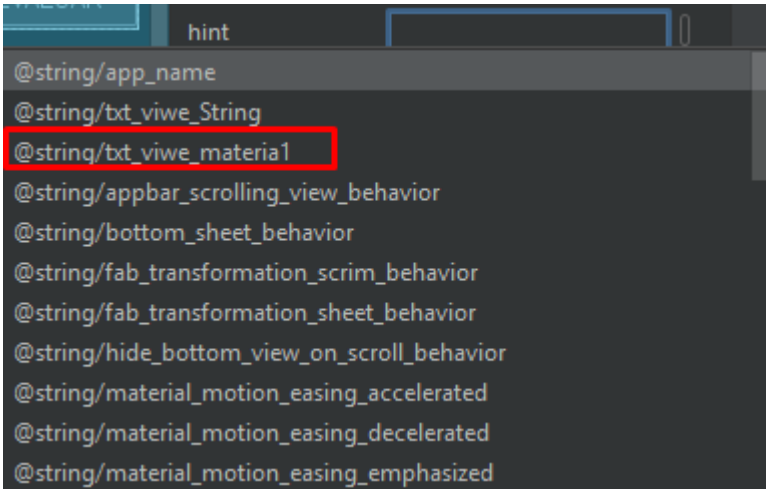


Así que volvemos a el archivo xml y ponemos el texto de Resultado

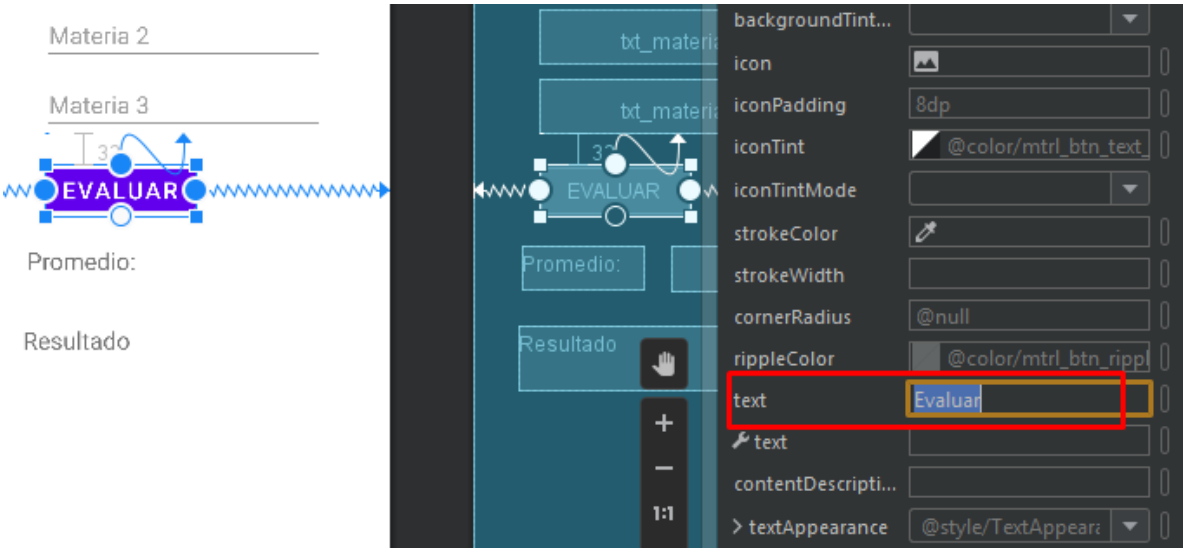
Ahora hagámoslo para un elemento Text pero con contenido en el apartado hint, como materia 1
<string name="txt_viwe_materia1">Materia1</string>
Vamos al diseño y hacemos lo mismo con la diferencia que ahora en lugar de usar el comando CTRL+BarraDeSpacio en Text iremos donde tenemos hint



Y asignamos la variable que acabamos de crear



Y listo hacemos las etiquetas del resto de componentes, en el botón se realiza igual solo en la parte de que le corresponde



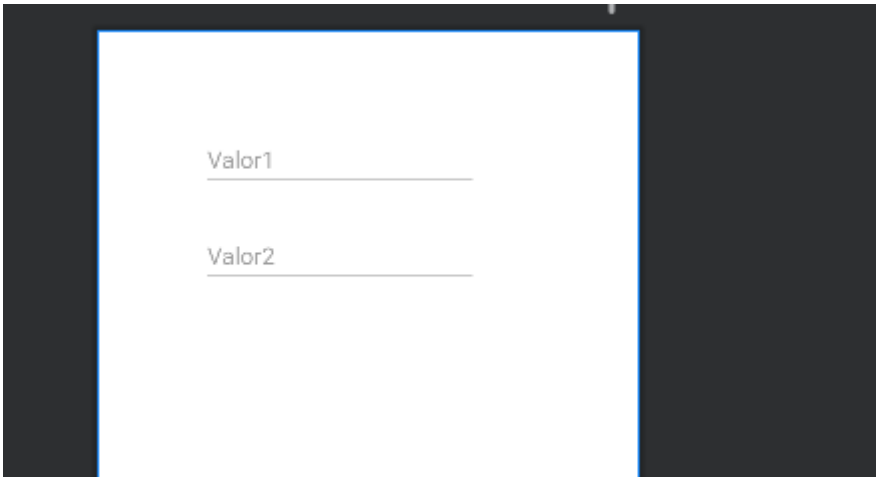
Codigo

```
<resources>
    <string name="app_name">Promedio</string>
//textview
    <string name="txt_viwe_String">Resultado</string>
    <string name="txt_viwe_String2">Promedio:</string>
    <string name="txt_viwe_String3"></string> // este textview no tiene
nada pero lo ponemos
//boton
    <string name="txt_viwe_boton">Evaluar</string>
// materias
    <string name="txt_viwe_materia1">Materia 1</string>
    <string name="txt_viwe_materia2">Materia 2</string>
    <string name="txt_viwe_materia3">Materia 3</string>
</resources>
```

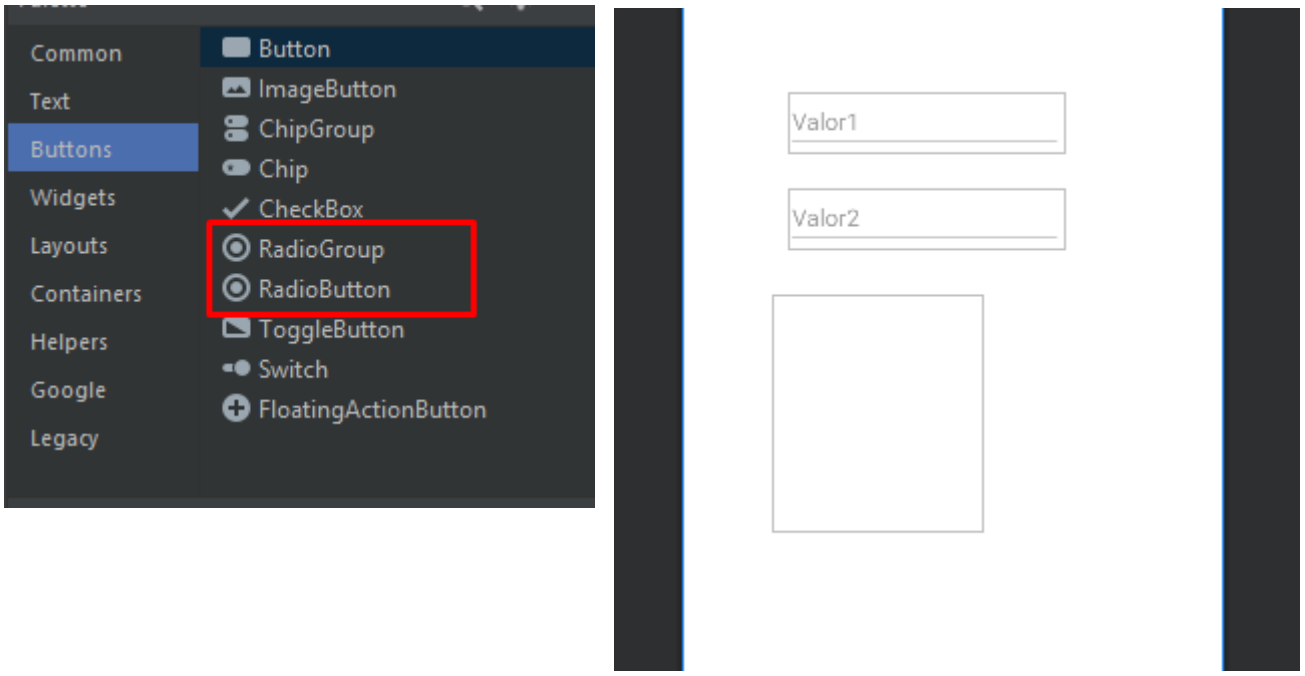
12) Controles RadioGroup y RadioButton(parte grafica)

Los controles Radio group y radio button permiten al usuario el elegir entre distintas opciones para este caso vamos a crear una calculadora que pueda sumar y restar, pero con un solo botón calcular

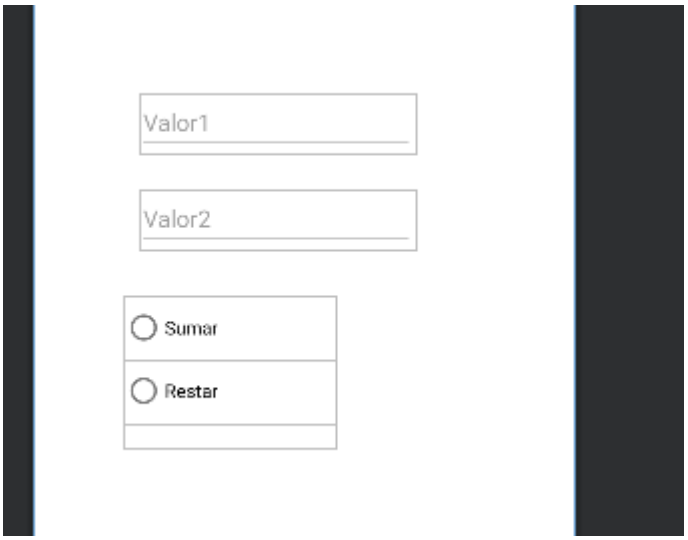
Abrimos un nuevo proyecto, agregamos 2 texview numéricos



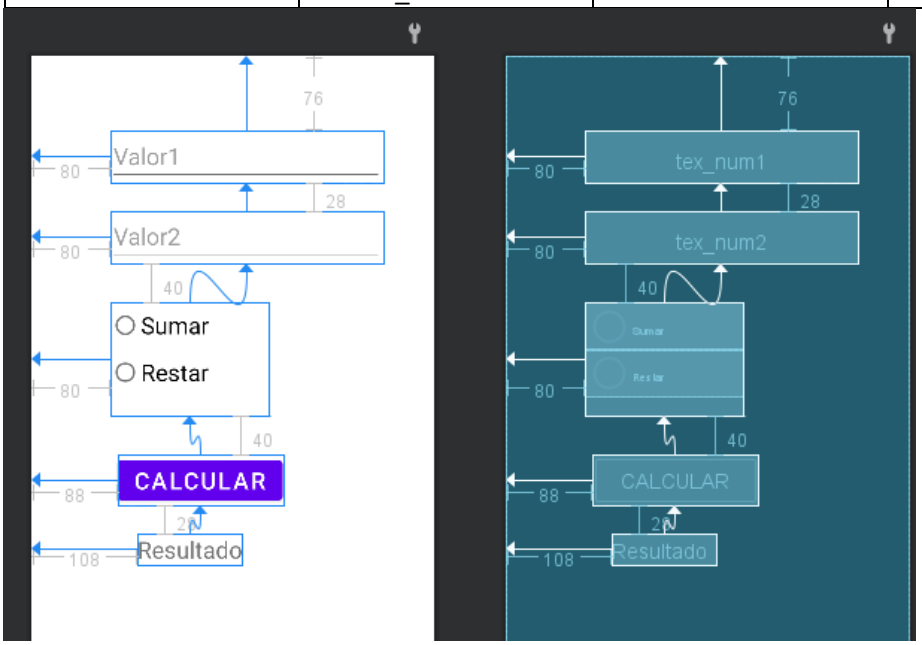
Y ahora colocamos el Radiogroup, ajustamos su tamaño



Ahora agregamos los 2 radio button y asignamos nombres con ayuda del archivo string.xml como en el tema pasado



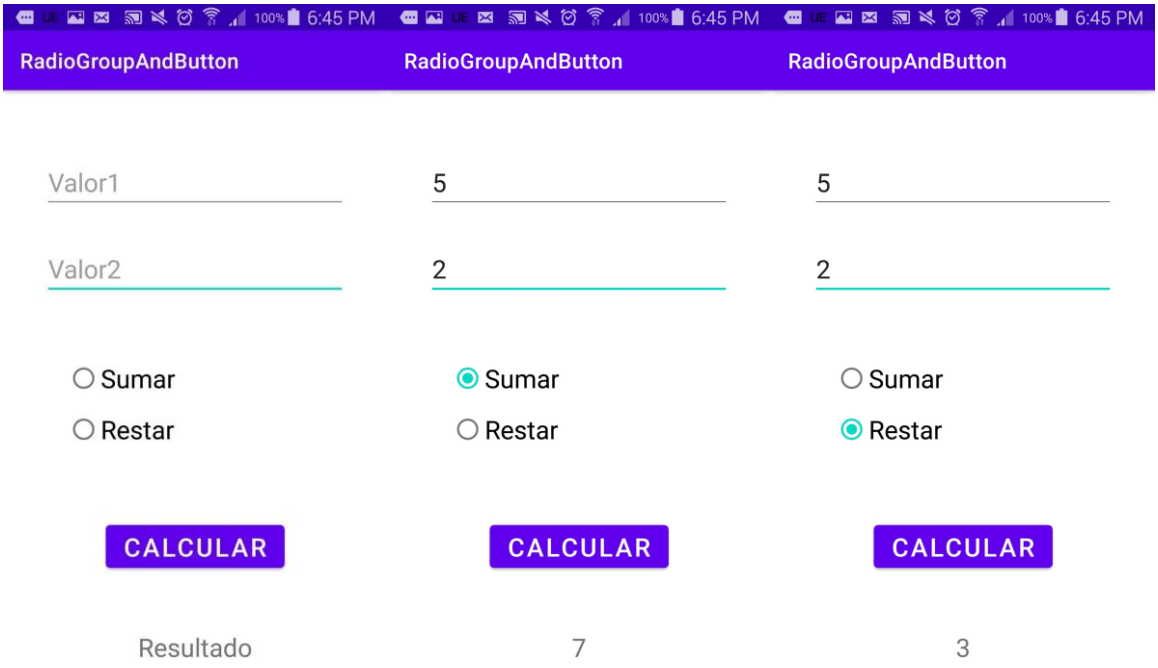
Elemento	Id	Hint	Text
Textview	tex_num1	Valor1	--
Textview	tex_num2	Valor	--
Textview	txt_resultado	--	Resultado
Radio button	radioB_suma	--	Sumar
Radio button	radioB_resta	--	Restar
Botton	Button_1	---	Calcular



13) Controles RadioGroup y RadioButton(parte logica)

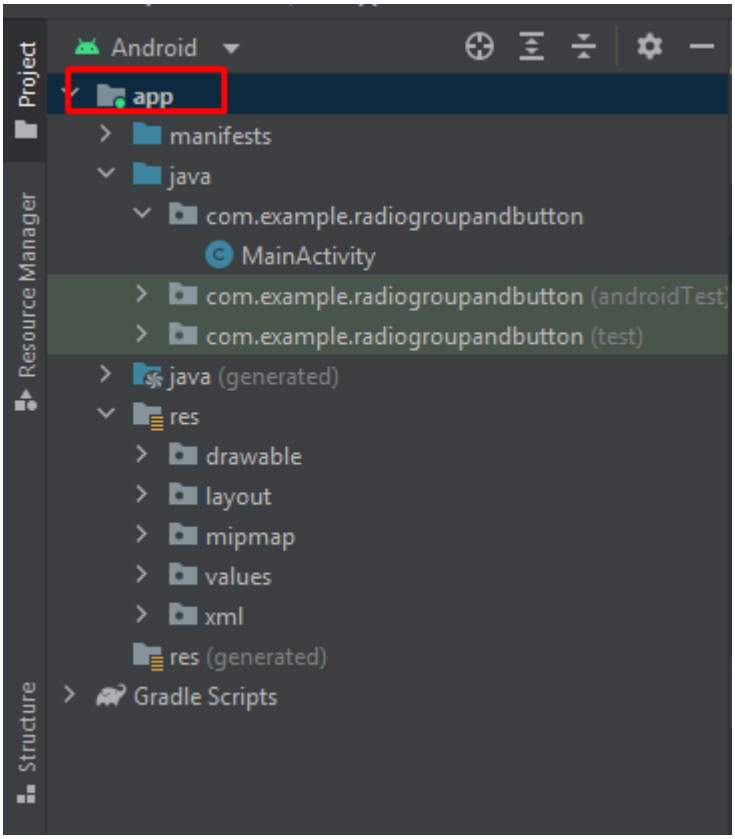
```
package com.example.radiogroupandbutton;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
private EditText et1,et2;
private TextView tv1;
//declaramos variables para los radiobutton
private RadioButton rb1, rb2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
       et1=(EditText) findViewById(R.id.tex_num1);
       et2=(EditText) findViewById(R.id.tex_num2);
       tv1=(TextView) findViewById(R.id.txt_resultado);
       rb1 = (RadioButton) findViewById(R.id.radioB_suma);
       rb2=(RadioButton) findViewById(R.id.radioB_resta);
    }

    //metodo calcular
    public void Calcular(View view1){

        String valor1_string =et1.getText().toString();
        String valor2_string =et2.getText().toString();

        int valor1_int=Integer.parseInt(valor1_string);
        int valor2_int=Integer.parseInt(valor2_string);
//usamos if para indicar el funcionamiento del boton
//nombreRadioButon.isChecked()==true  con esto decimos que esta
seleccionado y activamos el if
        if(rb1.isChecked()==true){
            int resultado =valor1_int +valor2_int;
            String resultado_string = String.valueOf(resultado);
            tv1.setText(resultado_string);
        }if(rb2.isChecked()==true){
            int resultado =valor1_int - valor2_int;
            String resultado_string = String.valueOf(resultado);
            tv1.setText(resultado_string);
        }
    }
}
```

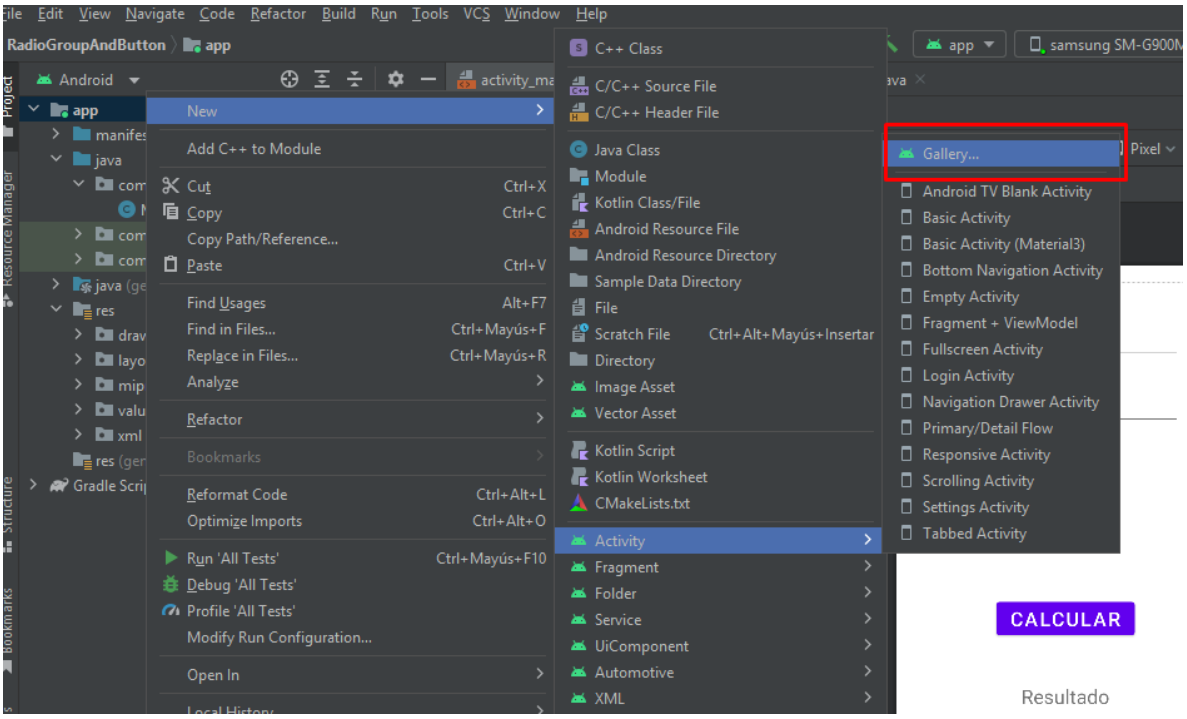
Resultado


6:45 PM
RadioGroupAndButton
Valor1
Valor2
Sumar
Restar
CALCULAR
Resultado
5
2
7
6:45 PM
RadioGroupAndButton
5
2
3
6:45 PM
RadioGroupAndButton

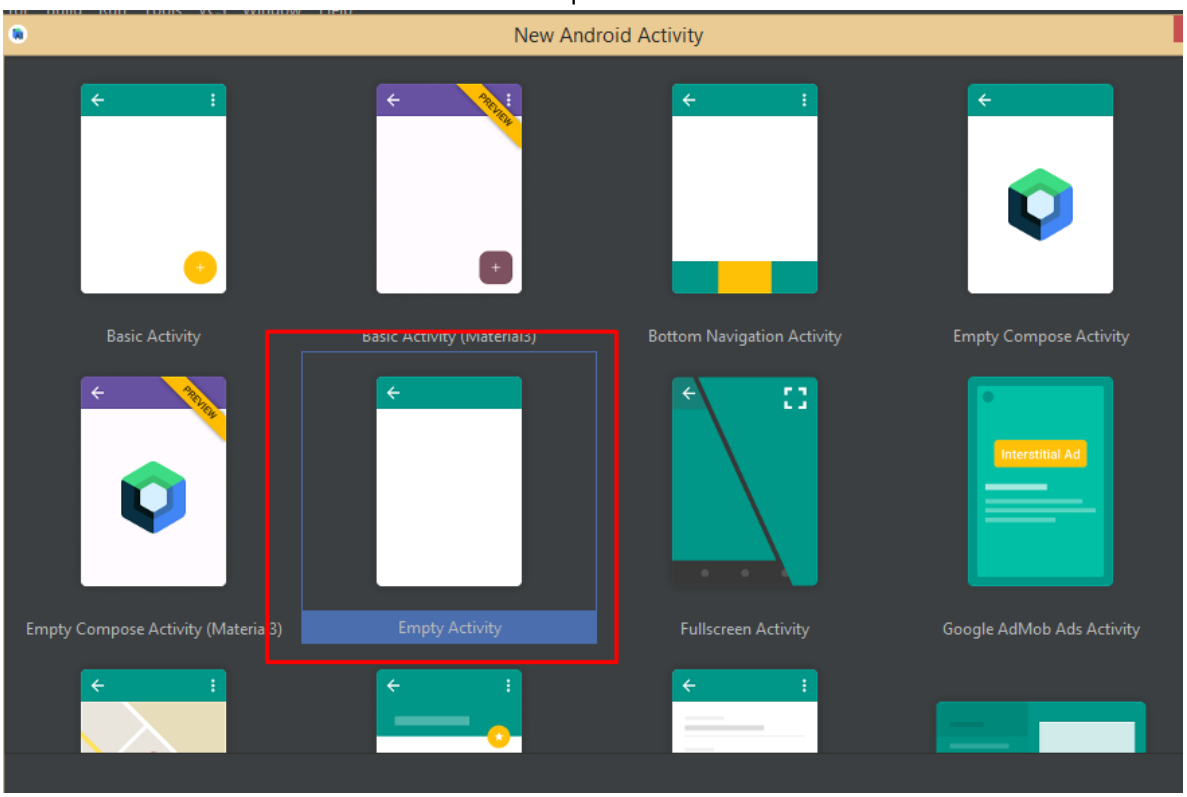
14) Pasar de un activity a otro (parte grafica)

Usaremos el ejercicio pasado, primero crearemos una nueva activity para ellos clic derecho en app en la lista de Project



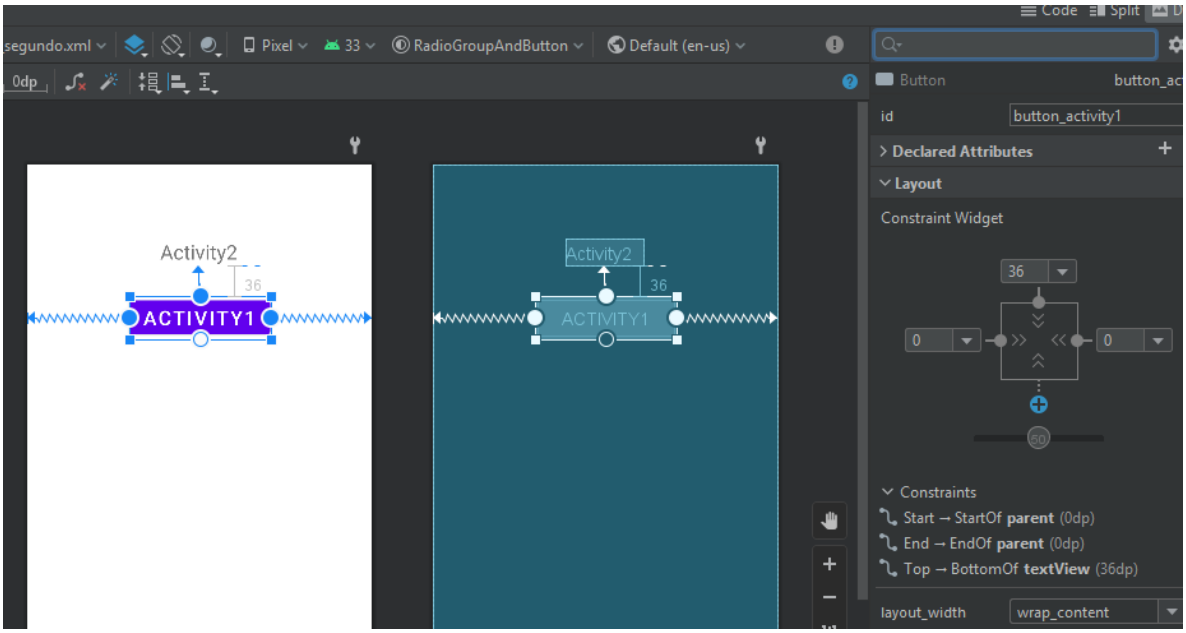


Ahora en la ventana seleccionamos la ventana que necesitamos

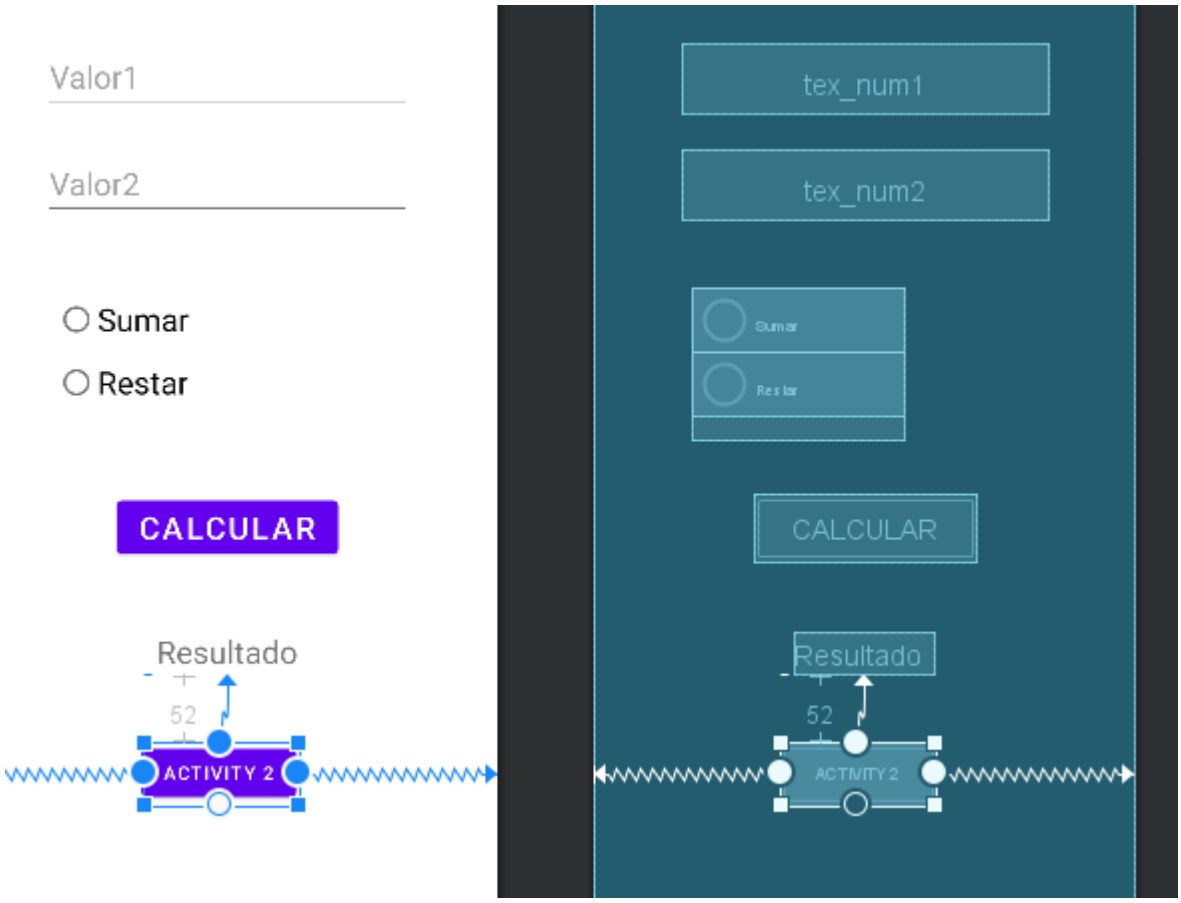


Colocamos el nombre a nuestra activity en este cado SegundoActivity

Agregaremos un textview que diga Activity 2 y un botón que se llame Activity1 con un id que sea button_activity1



Regresamos al MainActivity y también agregamos un nuevo botón, que se llame Activity 2 y con id que sea button_activity2



15) Pasar de un activity a otro (parte logica)

Codigo activity_segundo

```
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;

public class SegundoActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_segundo);
    }
    //metodo para moverte entre activity
    public void moverAct1(View view1){
        //creamos un objeto tipo intent
    //Intent nombreObj = new Intent (this , nombreActivity.class);
        // el this los usamos para indicar que es este activity
        Intent act1 = new Intent(this, MainActivity.class);
        //necesitamos dar inicio al activiy
        startActivity(act1);
    }

}
```

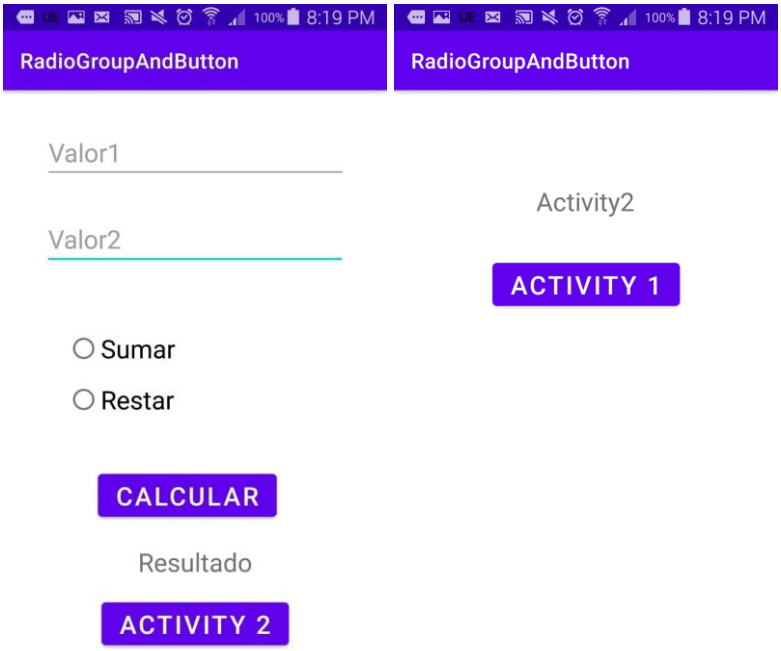
Codigo MainActivity

```
package com.example.radiogroupandbutton;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
private EditText et1,et2;
private TextView tv1;
//declaramos variables para los radiobutton
private RadioButton rb1, rb2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
       et1=(EditText) findViewById(R.id.tex_num1);
       et2=(EditText) findViewById(R.id.tex_num2);
       tv1=(TextView) findViewById(R.id.txt_resultado);
       rb1 = (RadioButton) findViewById(R.id.radioB_suma);
       rb2=(RadioButton) findViewById(R.id.radioB_resta);
    }
```

```
    //metodo calcular
    public void Calcular(View view1){

        String valor1_string =et1.getText().toString();
        String valor2_string =et2.getText().toString();

        int valor1_int=Integer.parseInt(valor1_string);
        int valor2_int=Integer.parseInt(valor2_string);
//usamos if para indicar el funcionamiento del boton
//nombreRadioButon.isChecked()==true  con esto decimos que esta
seleccionado y activamos el if
        if(rb1.isChecked()==true){
            int resultado =valor1 int +valor2 int;
            String resultado_string = String.valueOf(resultado);
            tv1.setText(resultado_string);
        }if(rb2.isChecked()==true){
            int resultado =valor1_int - valor2_int;
            String resultado_string = String.valueOf(resultado);
            tv1.setText(resultado_string);
        }
    }
    //metodo para moverte entre activity
    public void moverAct2(View view1){
        //creamos un objeto tipo intent
        //Intent nombreObj = new Intent (this , nombreActivity.class);
        // el this los usamos para indicar que es este activity
        Intent act2 = new Intent(this, SegundoActivity.class);
        //necesitamos dar inicio al activiy
        startActivity(act2);
    }
}
```

Le asignamos al boton su función en cada xml

Resultado



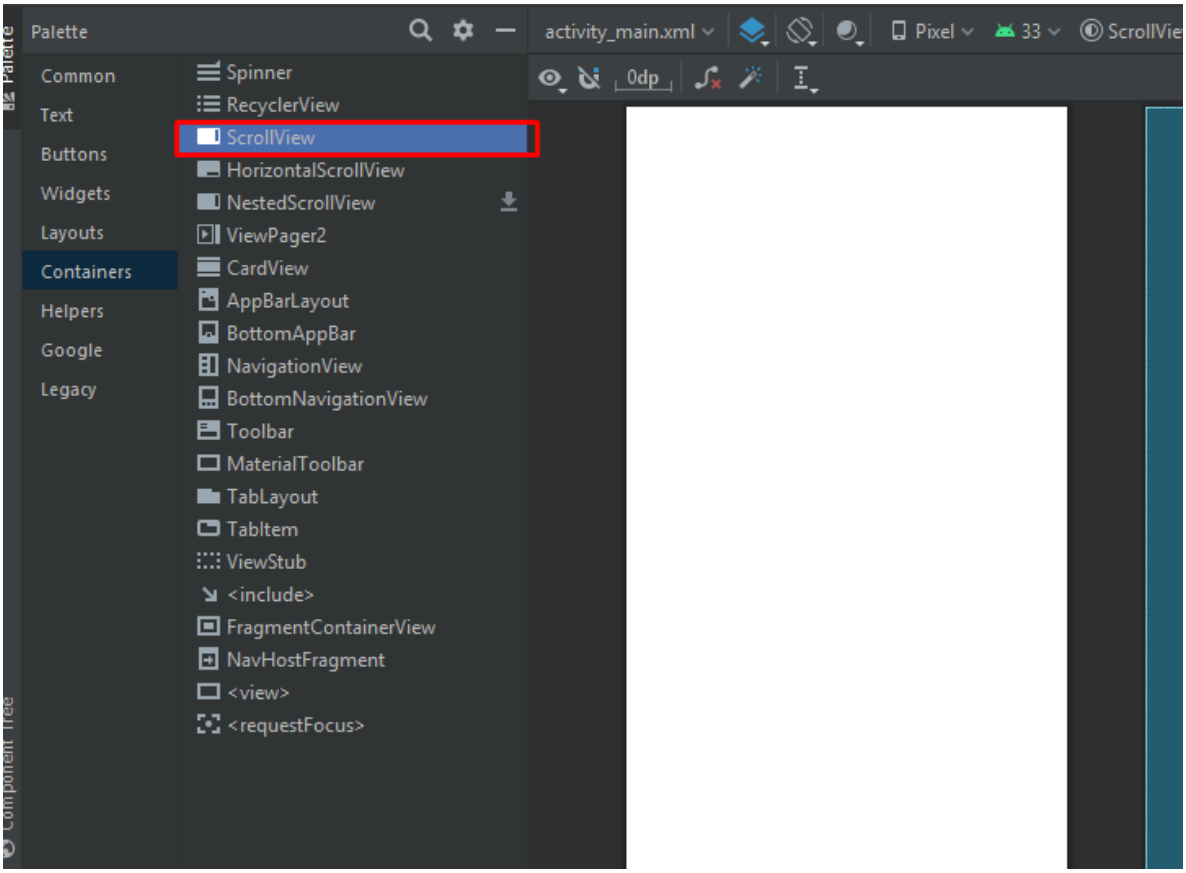
Aquí en imágenes obviamente no se ve lo que es el cambio entre ventanas, pero podemos observar eso al correr el programa, ademas podemos ver que estando en el activity 2 si usamos el boton de regresar

Regresamos a la MainActivity

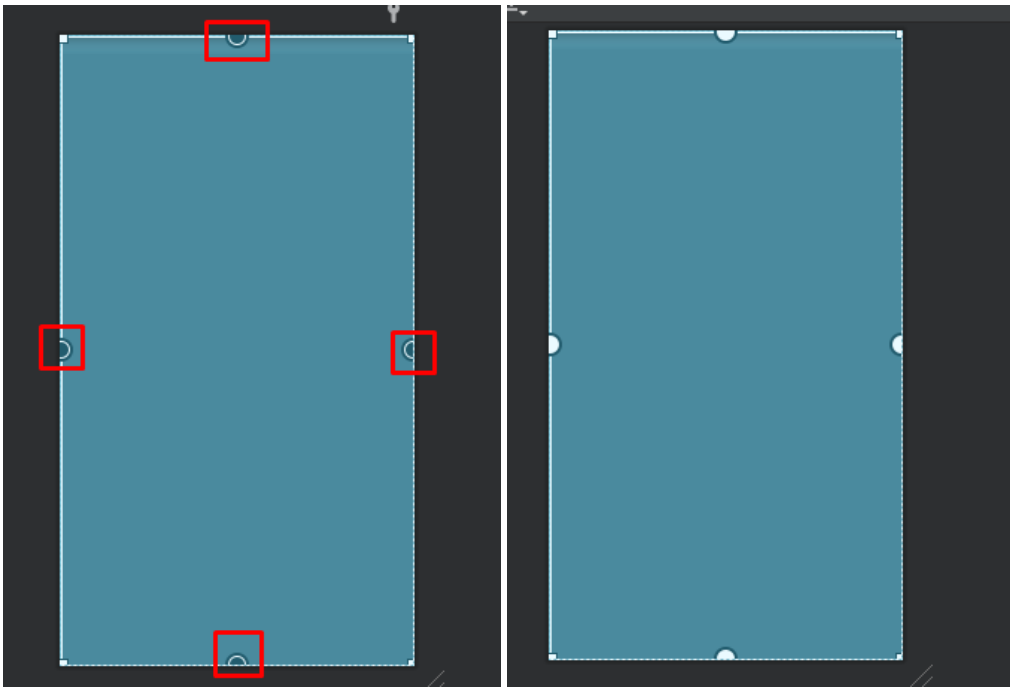
16) ScrollView - desplazar vista (parte grafica)/ Imagen Button

Esta función nos permite tener una cantidad de elementos que superen la vista del dispositivo Realizaremos un ejercicio para verlo más a detalle, crearemos una app que nos muestre una lista de frutas, para ello descarga imágenes de frutas y guárdalas en una carpeta, asegúrate de que las imágenes no contengan en su nombre mayúsculas al iniciar ni ningún otro carácter especial (@!#$%&) o que contenga la ñ en el nombre, y de preferencia tipo PNG (sin fondo) para el perfecto uso de este ejemplo o bien descarga las desde este link:
https://drive.google.com/file/d/1babuH4SIC9S1FmHSM4n1240_xYx03DGK/view?usp=sharing
Descomprime el archivo y comencemos.

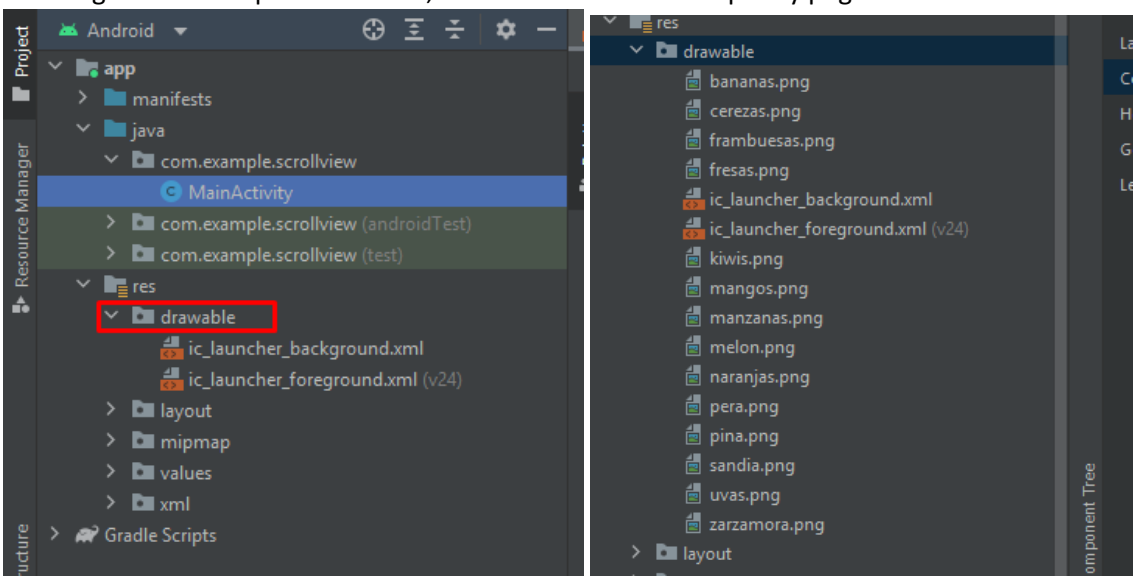
Crearemos un nuevo proyecto le pondremos ScrollView y crearemos una activity vacia. Agregamos un elemento ScrollView



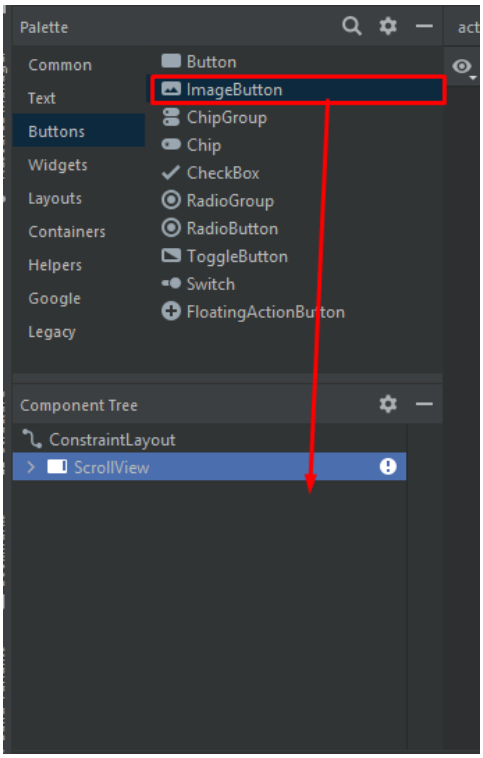
Podemos ver que al momento de soltarlo en la vista ocupa de inmediato toda la pantalla disponible de diseño, ahora nos vamos a blupritn y ajustamos los bordes, cada uno con su respectivo lado



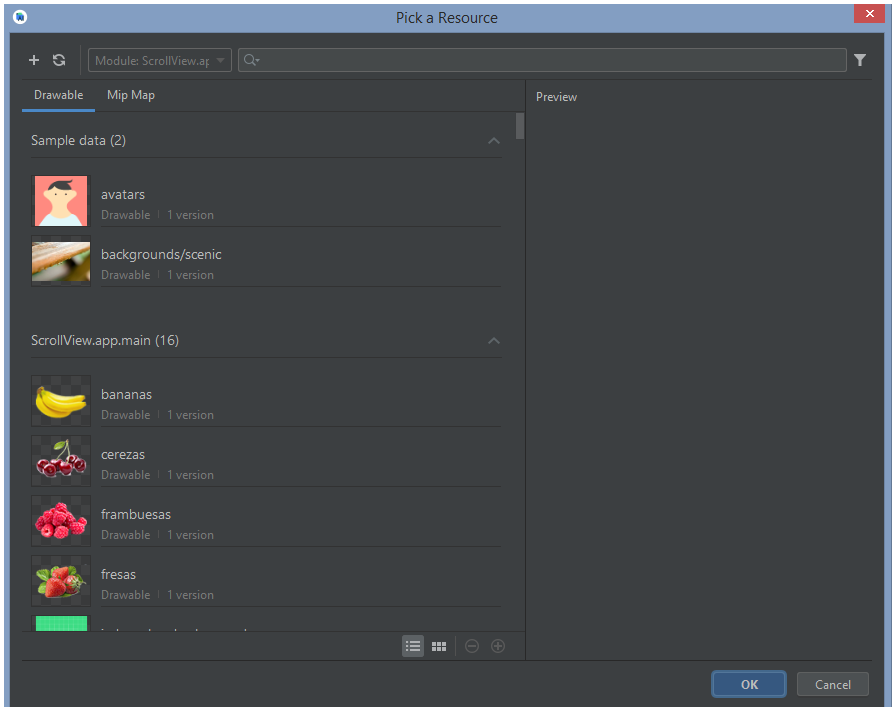
Podemos observar que hasta que no anclamos los bordes estos se quedan de colo azul oscuro, ahora vamos a la carpeta donde tenemos las imágenes de las frutas y las copiamos todas, posteriormente nos dirigimos a la carpeta drawable, clic derecho sobre la carpeta y pegamos



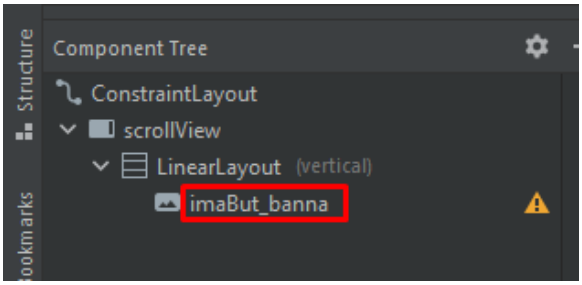
A partir de este momento estas imágenes seran parte de nuestro proyecto y no improta que las borremos de la carpeta donde se descargaron.

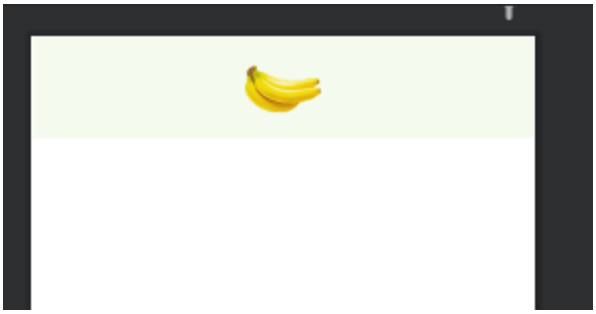


Ahora agregamos un imagen button, para ello seleccionamos y arrastramos a Component Tree, nos saldra otra ventana para elegir el recurso (pick a resource)y elegimos la fruta que queramos en este caso ponde banana y damos en ok

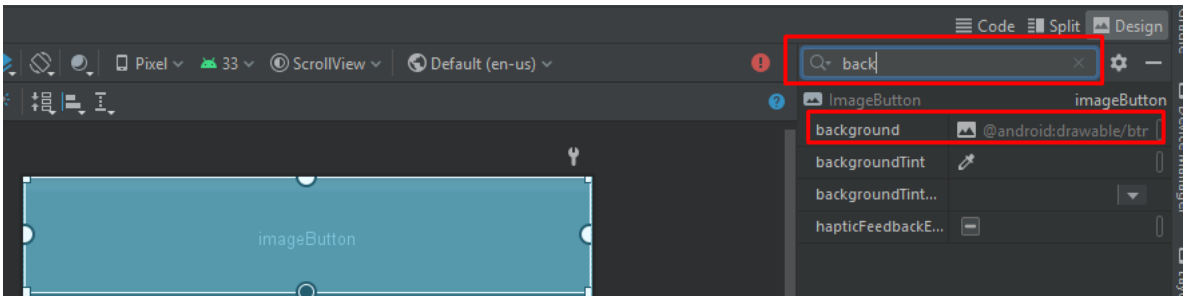


Nota importante los botnes deben estar dentro del LinearLayout, podemos cambiar eso en el Component tree solo arrastrando el elemento

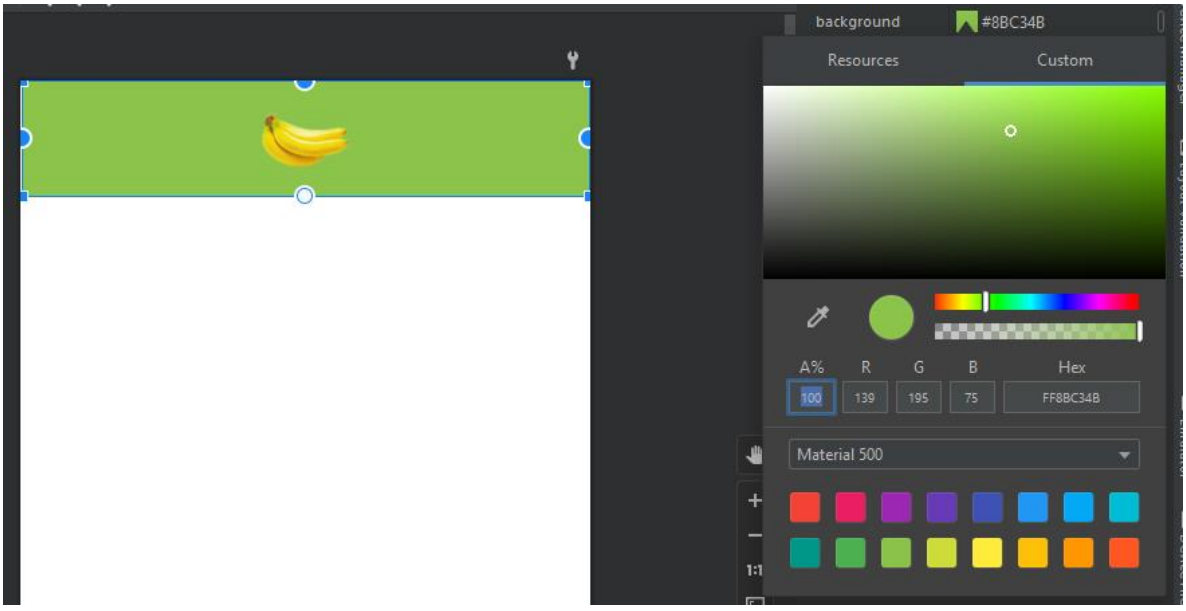




Posteriormente vamos a modificar en atributos, podemos ayudarnos del bsucador y ponemos background



Y asignamos un color

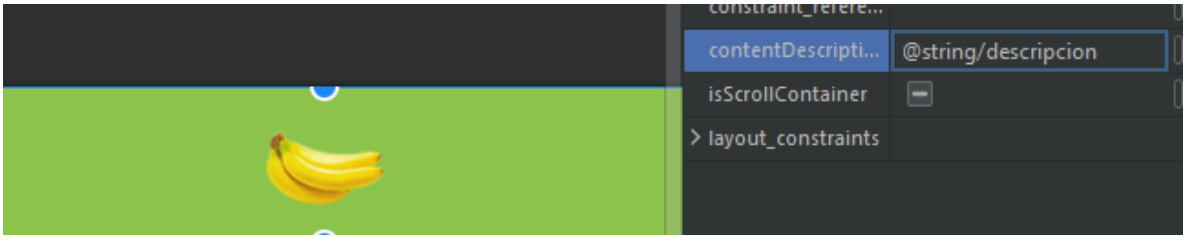


No olvidemos colocarle un id al boton

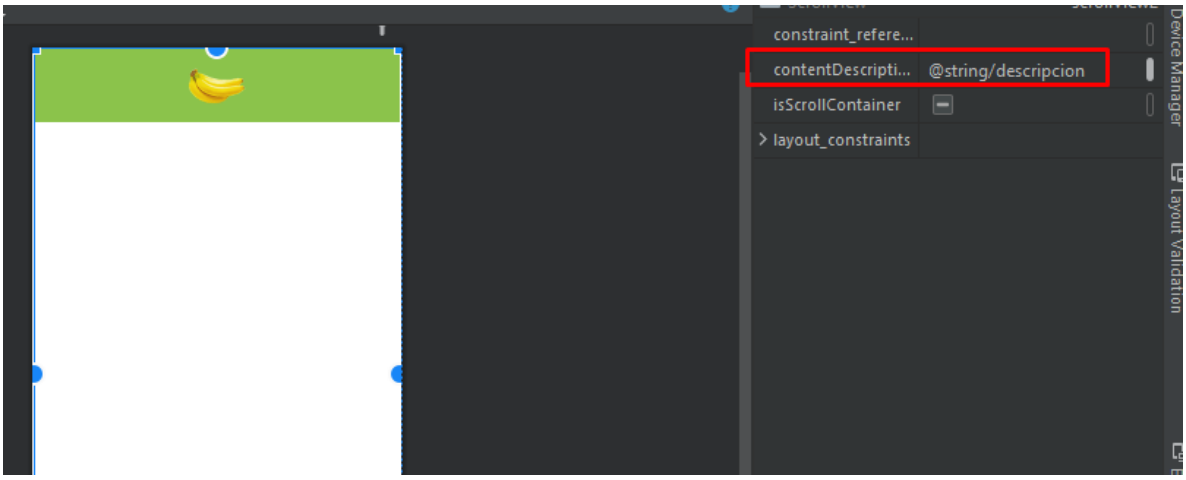
Ahora android studio nos marcara error mientras no coloquemos un contentDescription abrimos nuestro archivo String que trabajamos en Hardcode string shoul use string resource , y escribimos el siguiente codigo

<string name="descripcion">Descripcion</string>

Regresamos a los atributos del boton y asignamos el contentDescription

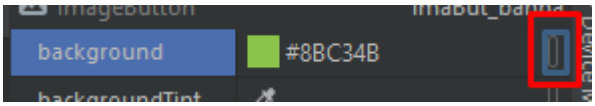


Hacemos lo mismo con el scrollview

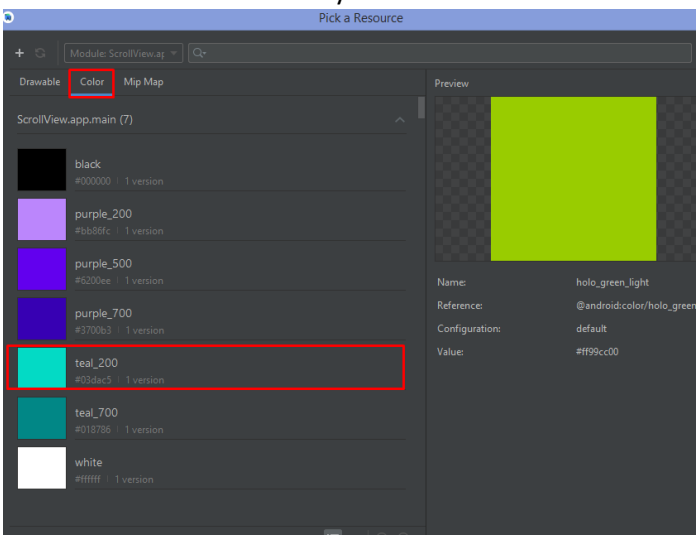


Como podemos ver no necesitamos hacer otra declaracion en el archivo string este mismo podemos usar para todos los botones

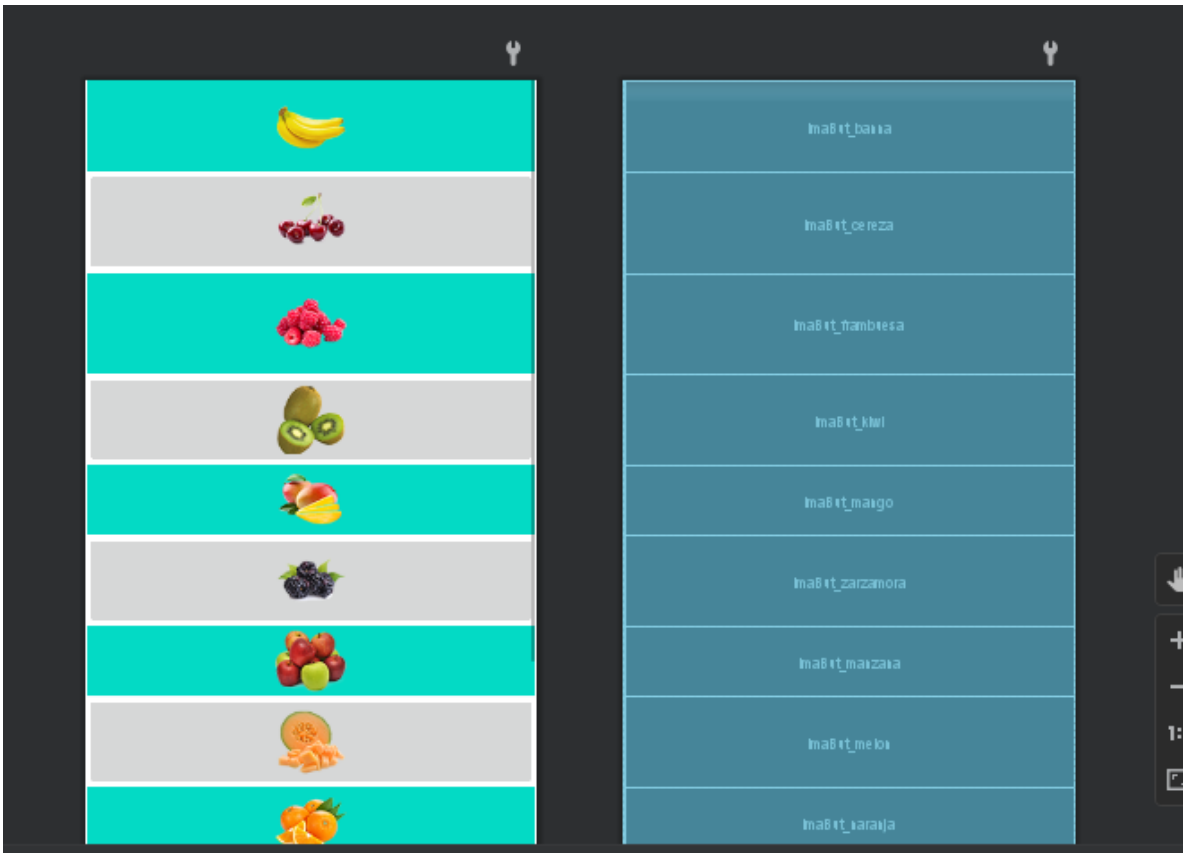
Otra forma de cambiar el color es en background seleccionar la barra que esta a la derecha



En este caso iremos a color y lo cambiaremos



Agregamos los demas botones y hacemos lo mismo, definimos id, contentDescriprion y el background. **Nota al poner el id de piña, evitemos la ñ y pongamos pina**



Como seguramente ya te diste cuenta en este punto, tambien puedes moverte con el scroll del mouse en las vistas de diseño y bluprint

17) ScrollView - desplazar vista (parte lógica)

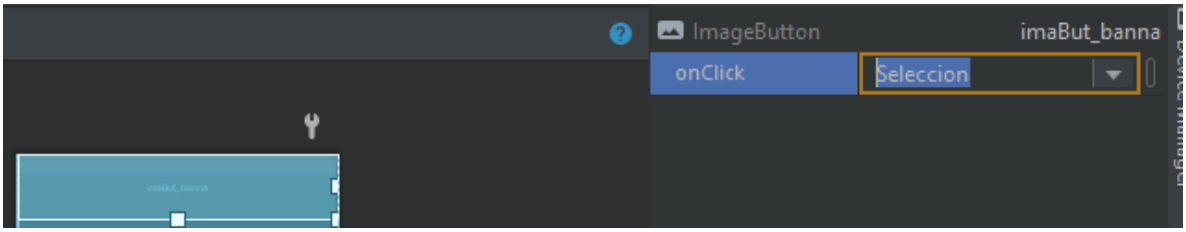
Si bien hasta aquí el funcionamiento de nuestra app (es decir que nos podamos desplazar hacia abajo en la pantalla) ya está completo, agreguemos unas funciones ayudándonos de switch y toast

Codigo

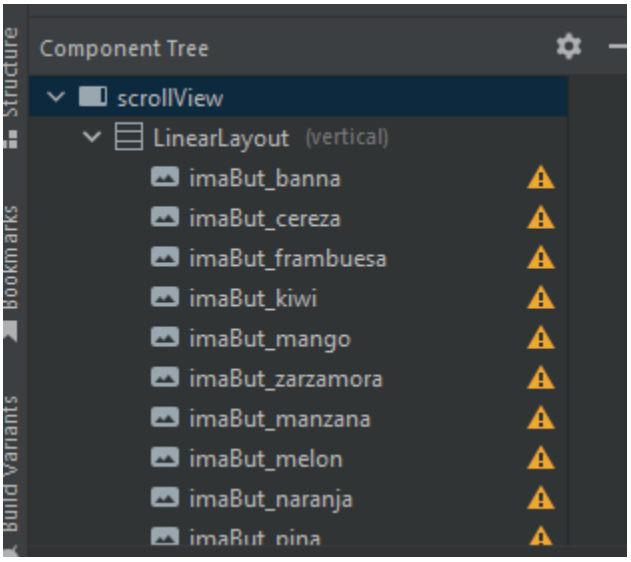
```
package com.example.scrollview;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

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

    public void Seleccion(View view1){
```

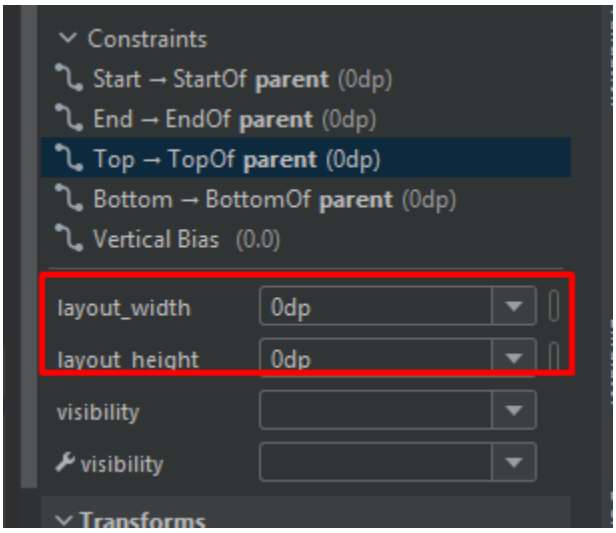
```
        switch (view1.getId()){ //con este decimos que dependiendo el
boton active por el id
            case R.id.imaBut_banna: // observese que no tuvimos que
declarar variables ára ajecutar
                // el metodo switc
                Toast.makeText(this,"Son
bananas",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_cereza:
                Toast.makeText(this,"Son
cerezas",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_frambuesa:
                Toast.makeText(this,"Son
frambuesas",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_kiwi:
                Toast.makeText(this,"Son
kiwis",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_mango:
                Toast.makeText(this,"Son
mangos",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_manzana:
                Toast.makeText(this,"Son
manzanas",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_melon:
                Toast.makeText(this,"Son
melon",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_naranja:
                Toast.makeText(this,"Son
narnajas",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_pina:
                Toast.makeText(this,"Son
piñas",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_pera:
                Toast.makeText(this,"Son pera",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_uvas:
                Toast.makeText(this,"Son uvas",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_zarzamora:
                Toast.makeText(this,"Son
zarzamora",Toast.LENGTH_LONG).show();
                break;
            case R.id.imaBut_sandia:
                Toast.makeText(this,"Son
sandias",Toast.LENGTH_LONG).show();
                break;
        }
    }
}
```

Recordemos asignarle el método a cada botón

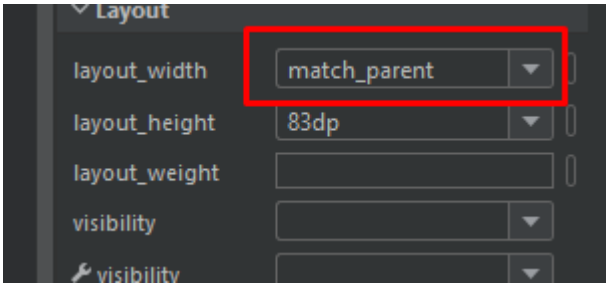


Si tienes problemas con el ScrollView es decir que no se puede desplazar por completo vamos a component tree y seleccionamos nuestro scrollView, ahora nos aseguramos que en atributos en layout_width y layout_height tengamos 0dp





Si algun boton no mantiene su tamaño al girar la pantalla nos aseguramos que sus atributos estén asi



Resultados