2. APP CODEBUTTON

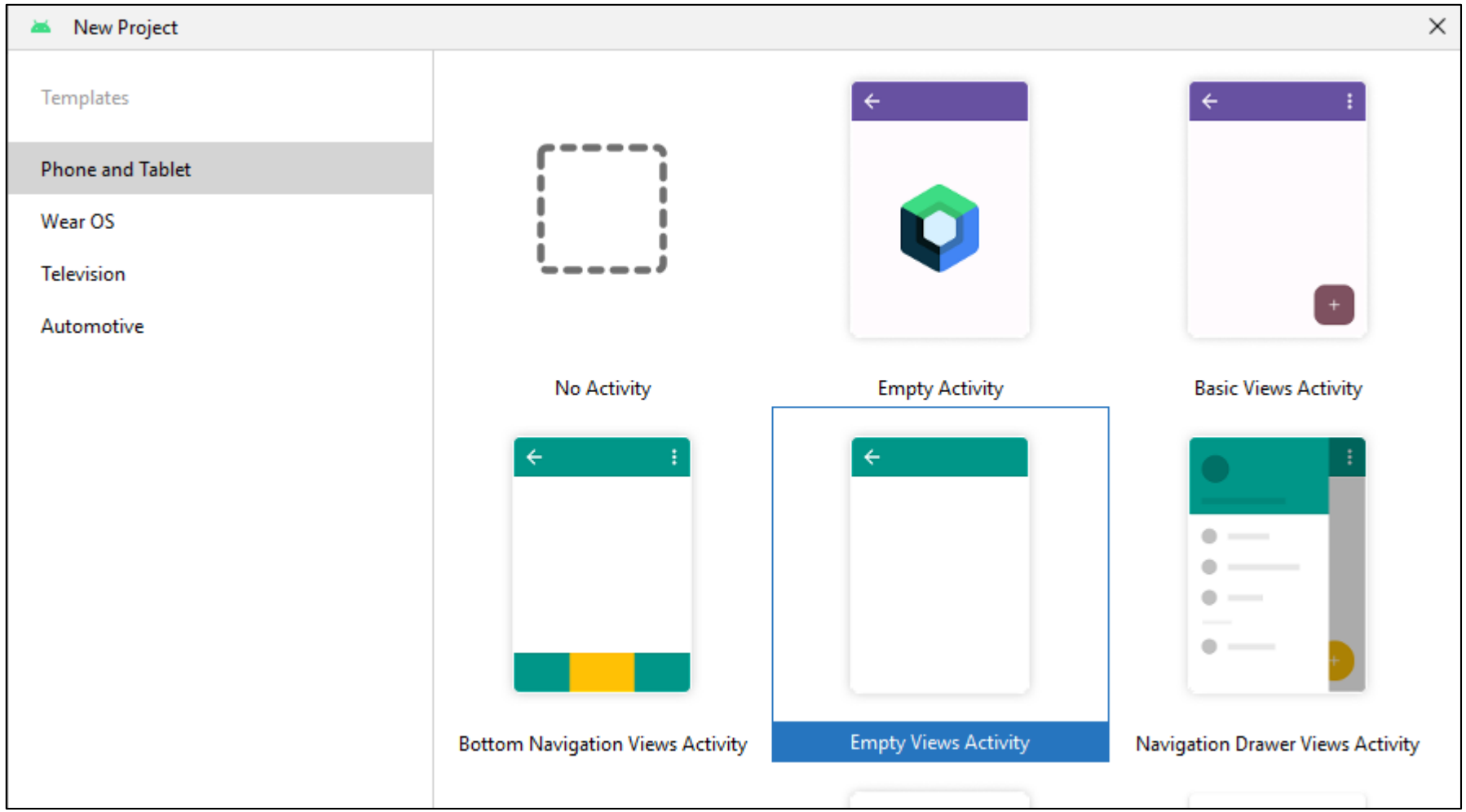
Objetivos

En esta primera aplicación veremos las diferentes formas que presenta Android de asociar el código del evento clic de un botón. Existen dos formas básicas, pero se pueden extender a 4, en función de donde se implemente la clase Listener que recoge el clic:

1) Mediante la definición de un objeto anónimo de la clase OnClickListener, y asociando el objeto con el botón dentro de OnCreate de MainActvity
2) Implementando la interfaz OnClickListener en el MainActivity.java.
3) Creando una clase aparte que implemente el listener OnClickListener
4) Enlazando una función con argumentos View a través de la propiedad onClick XML del botón

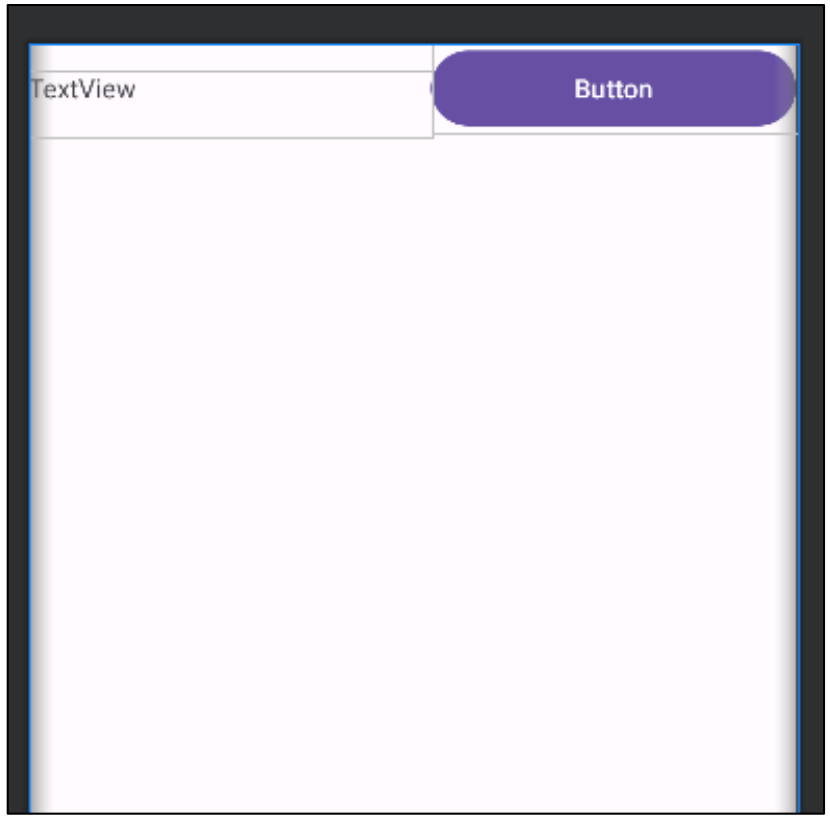
2. APP CODEBUTTON

Paso 1. Creamos una nueva aplicación Android, mediante la opción Empty Views Activity



2. APP CODEBUTTON

Paso 2. En el activity_main.xml primero pasamos el layout principal de ConstraintLayout (layout por defecto) a LinearLayout, con orientación horizontal (por facilidad). A continuación insertamos un textview y un button.




```xml
<?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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="111dp"
        android:layout_height="35dp"
        android:layout_weight="1"
        android:text="TextView" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Button" />
</LinearLayout>
```

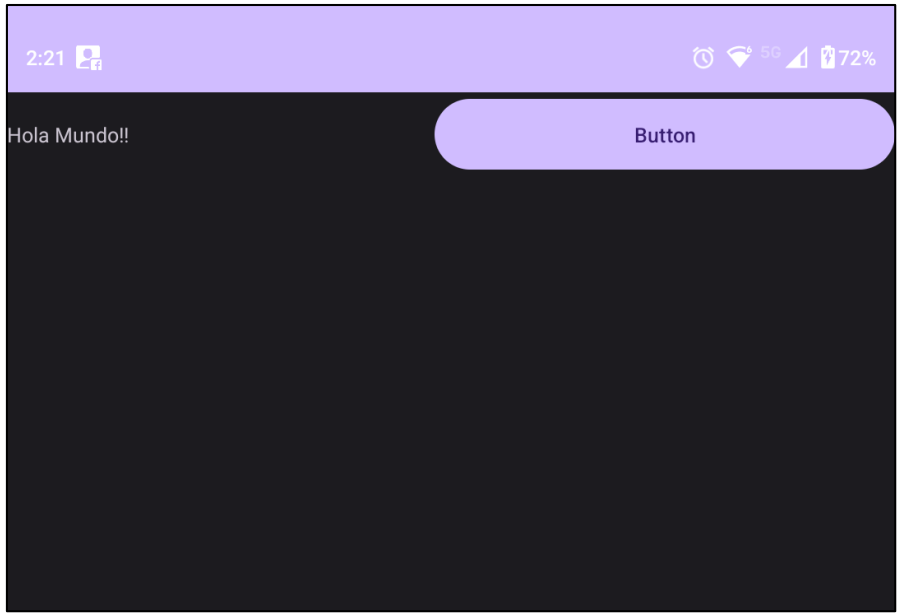
2. APP CODEBUTTON

Paso 3. En MainActivity .java definimos dos atributos de la clase, un Button y un TextView y los enlazamos con los elementos xml creados anteriormente mediante la función findViewById, dentro OnCreate de MainActivity

```java
package com.example.code_boton;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.Button;
import android.widget.TextView;

2 usages
public class MainActivity extends AppCompatActivity {

    1 usage
    private Button mybutton;
    1 usage
    private TextView mytext;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        this.mybutton=(Button)findViewById(R.id.button);
        this.mytext=(TextView)findViewById(R.id.textView);
    }
}
```

2. APP CODEBUTTON

Paso 4. Primera forma definición de código del botón. Mediante la definición de un objeto anónimo de la clase OnClickListener y asociación con su botón correspondiente dentro de OnCreate

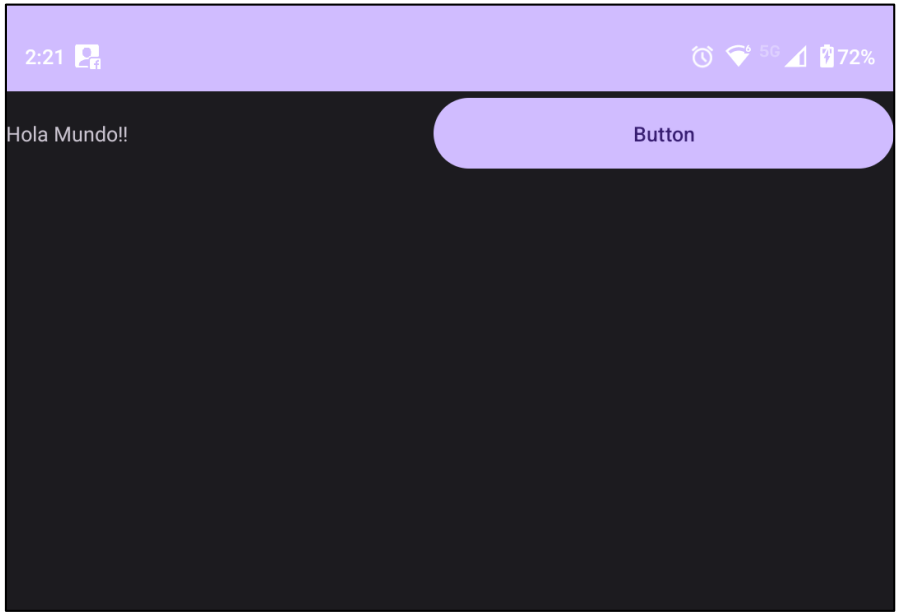


Forma mas usada, pero que tiene un consumo mayor de memoria

```java
package com.example.codebutton;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

2 usages
public class MainActivity extends AppCompatActivity {
    2 usages
    private Button mybutton;
    2 usages
    private TextView mytextview;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        this.mybutton = (Button)findViewById(R.id.button);
        this.mytextview = (TextView)findViewById(R.id.textView);
        this.mybutton.setOnClickListener(new View.OnClickListener(){
            @Override public void onClick(View view) { mytextview.setText("Hola Mundo!!"); }
        });

    }
}
```

2. APP CODEBUTTON

Paso 5. Segunda forma definición de código del botón. Implementando la interface OnClickListener en el MainActivity.java



El problema de este método es que comparte el método onClick con el código de todos los botones del layout. Se debe de separar mediante view.getId()

```java
package com.example.codebutton;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

2 usages
public class MainActivity extends AppCompatActivity implements View.OnClickListener{
    2 usages
    private Button mybutton;
    2 usages
    private TextView mytextview;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        this.mybutton = (Button) findViewById(R.id.button);
        this.mytextview = (TextView) findViewById(R.id.textView);
        this.mybutton.setOnClickListener(this);
        //this.mysecondbutton.setOnClickListener(this);
        }

    @Override
    public void onClick(View view){
        if(view.getId()== R.id.button) {
            mytextview.setText("Hola Mundo!!");
        }
        //else if(view.getId()=R.id.mysecondbutton){
        //    mytextview.setText("Hola second button!!");
        //}
    }
}
```

2. APP CODEBUTTON

Paso 6. Tercera forma definición del código del botón. Creando una clase aparte que implemente OnClickListener

```java
package com.example.codebutton;

import android.view.View;
import android.widget.TextView;

1 usage
public class MyCodeButton implements View.OnClickListener {
    2 usages
    TextView textview;
    1 usage
    public MyCodeButton(TextView mytextview){
        this.textview=mytextview;
    }
    @Override
    public void onClick(View view){
            textview.setText("Hola Mundo!!");
    }
}
```

La única pega es que se debería generar tantas clases que implemente el listener como botones tenga el layout.

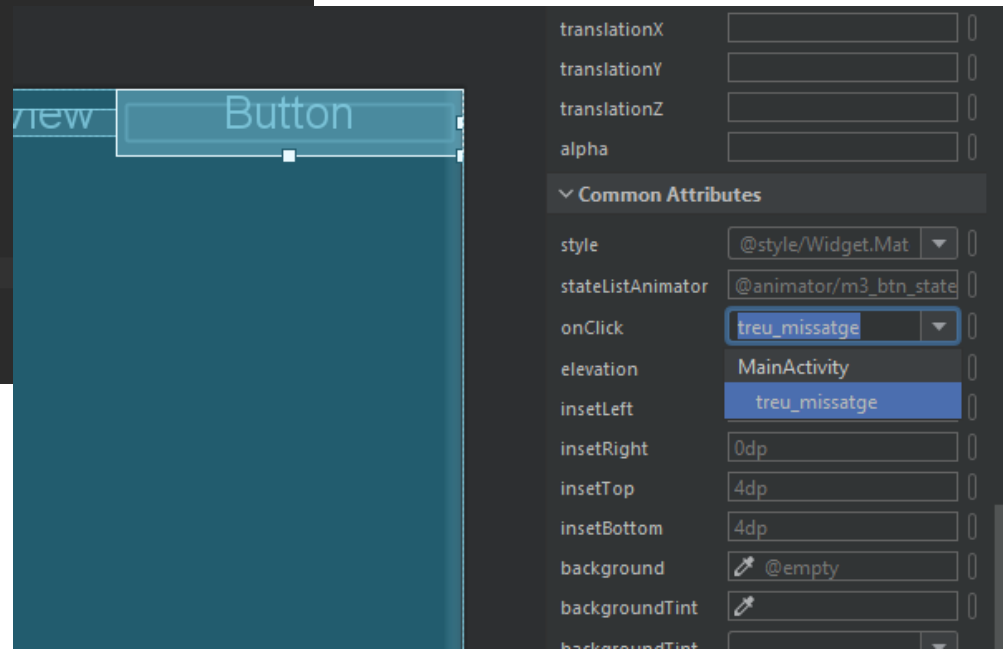
```java
package com.example.codebutton;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

2 usages
public class MainActivity extends AppCompatActivity {
    2 usages
    private Button mybutton;
    2 usages
    private TextView mytextview;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        this.mybutton = (Button) findViewById(R.id.button);
        this.mytextview = (TextView) findViewById(R.id.textView);
        this.mybutton.setOnClickListener(new MyCodeButton(mytextview));
    }
}
```

2. APP CODEBUTTON

Paso 7. Cuarta forma definición del código del botón. Enlazando una función con argumento View a través de la propiedad onClick del XML del botón

```java
package com.example.codebutton;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

2 usages
public class MainActivity extends AppCompatActivity {
    1 usage
    private Button mybutton;
    2 usages
    private TextView mytextview;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        this.mybutton = (Button) findViewById(R.id.button);
        this.mytextview = (TextView) findViewById(R.id.textView);
    }
    no usages
    public void treumissatge(View View){
        mytextview.setText("Hello world!!");
    }
}
```

```xml
<?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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:onClick="treu_missatge"
        android:text="Button" />
</LinearLayout>
```



Opción elegida junto con la primera

2. APP IMC

Objetivos

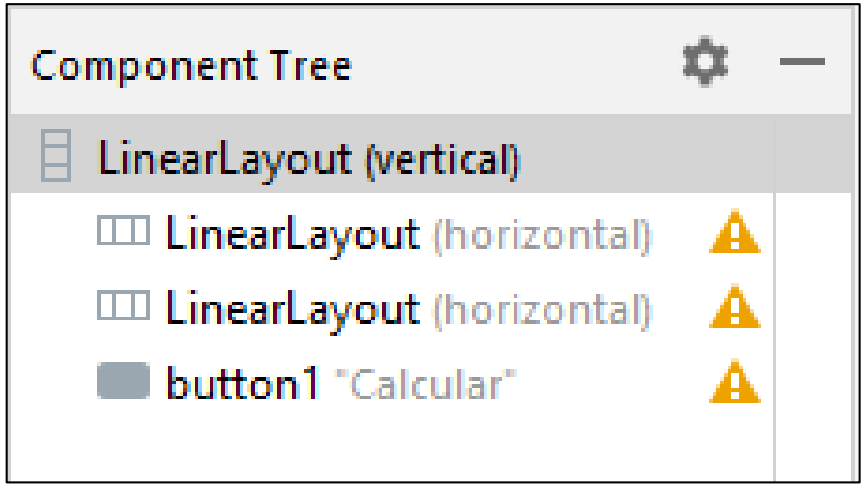
En esta segunda aplicación haremos la típica aplicación IMC Indice de Masa Corporal, que con los datos de la altura y el peso nos indica si estamos en línea o un poco gorditos

Aquí empezaremos a trabajar las diferentes formas de layout, en concreto veremos el funcionamiento del tipo LinearLayout. Veremos como enlazar dos layouts de tipo LinearLayout, uno con orientación vertical y el otro con orientación horizontal, de manera que nos permita el despliegue de nuestros controles de la forma deseada.

También introduciremos la estructura Toast que nos permite ver un mensaje en la parte media-baja de la pantalla que dura unos segundos

2. APP IMC

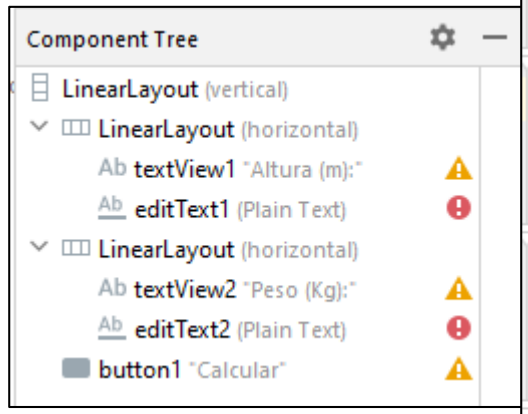
Paso 1. Declararemos el layout principal de tipo LinearLayout con orientación vertical. Agregaremos 2 LinearLayout con orientación horizontal y un botón, que se dispondrán verticalmente

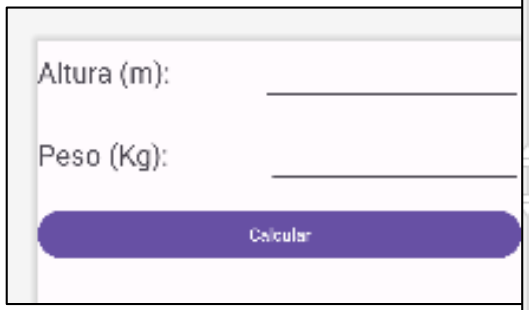



```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="70dp"
        android:orientation="horizontal"/>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="70dp"
        android:orientation="horizontal" />
    <Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Calcular" />
</LinearLayout>
```

2. APP IMC

Paso 2. Insertamos los textView y los editTexts para completar la aplicación






```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/ar
    xmlns:app="http://schemas.android.com/apk/res-auto"  xmlns:too
    android:layout_width="match_parent"  android:layout_height="ma
    android:orientation="vertical"     tools:context=".MainActivit
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="70dp"
        android:orientation="horizontal">
        <TextView
            android:id="@+id/textView1"
            android:layout_width="248dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Altura (m):"
            android:textSize="24sp" />
        <EditText
            android:id="@+id/editText1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:ems="10"
            android:inputType="text"
            android:textSize="24sp" />
    </LinearLayout>
```

```
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="70dp"
        android:orientation="horizontal" >
        <TextView
            android:id="@+id/textView2"
            android:layout_width="258dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Peso (Kg):"
            android:textSize="24sp" />
        <EditText
            android:id="@+id/editText2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"  android:ems="10"
            android:inputType="text"
            android:textSize="24sp" />
    </LinearLayout>
    <Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Calcular" />
</LinearLayout>
```

2. APP IMC

Paso 3. En MainActivity.java declaramos las 3 variables (1 botón y 2 edittexts) que enlazaremos con sus respectivos controles xml mediante findViewById en el evento onCreate de MainActivity

```
2 usages
public class MainActivity extends AppCompatActivity {
    1 usage
    private Button button;
    1 usage
    private EditText editText1;
    1 usage
    private EditText editText2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        button = (Button) findViewById(R.id.button1);
        editText1 = (EditText) findViewById(R.id.editText1);
        editText2 = (EditText) findViewById(R.id.editText2);
    }
}
```

2. APP IMC

Paso 4. Solo nos falta definir el cálculo que hace la aplicación al hacer click en el botón, una vez hemos introducido los datos de la altura y peso. Aquí utilizamos la función Toast, que durante 2 segundos nos da el resultado de nuestro IMC.

```
button.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View view) {
        if (editText1.equals("") || editText1.equals(""))
            Toast.makeText( context: MainActivity.this,  text: "Faltan paranmetros!",Toast.LENGTH_LONG).show();

        double altura = Double.parseDouble(editText1.getText().toString());
        double peso = Double.parseDouble(editText2.getText().toString());
        double imc= peso/(altura*altura);
        if (imc<18.5)
            Toast.makeText( context: MainActivity.this,  text: "Bajo peso corporal!",Toast.LENGTH_LONG).show();
        else if (imc<24.9)
            Toast.makeText( context: MainActivity.this,  text: "Normal!",Toast.LENGTH_LONG).show();
        else if (imc<29.9)
            Toast.makeText( context: MainActivity.this,  text: "Sobrepeso!",Toast.LENGTH_LONG).show();
        else
            Toast.makeText( context: MainActivity.this,  text: "Obeso!",Toast.LENGTH_LONG).show();
    }});
```

2. APP IMC

Paso 5. Ejemplo de ejecución de la aplicacion