

Desarrollo de Aplicaciones para Dispositivos Móviles

Ingeniero Walter Medina
walter.medina@correounivalle.edu.co



Componentes Layouts:

Los layouts son estructuras que definen la disposición y organización de los elementos de la interfaz en una aplicación. Los layouts determinan cómo se colocan y distribuyen los elementos, como botones, texto, imágenes, etc., en la pantalla del dispositivo.

Algunos tipos de Layouts:

- **LinearLayout**
- **RelativeLayout**
- **ConstraintLayout**
- **TableLayout**
- **GridLayout**



LinearLayout

Un LinearLayout es un tipo de vista contenedor que organiza sus elementos secundarios en una única dirección: horizontalmente o verticalmente. Es uno de los tipos más básicos de layouts disponibles en Android y se utiliza para colocar vistas (como botones, texto, imágenes, etc.) en una secuencia lineal.

```

10      <LinearLayout
11          android:layout_width="match_parent"
12          android:layout_height="match_parent"
13          tools:context=".MainActivity"
14          android:padding="16dp"
15          android:orientation="vertical">
16
17          <Button...>
23
24          <Button...>
30
31          <Button...>
37          <Button...>
43          <View...>
48
49          <LinearLayout...>
66
67          <LinearLayout...>
92
93          <LinearLayout...>
118
119      </LinearLayout>

```

RelativeLayout

Es un tipo de vista de contenedor que permite organizar los elementos de la interfaz en relación unos con otros o con el contenedor padre. Esto significa que puedes posicionar los elementos en función de otros elementos, como alinearlos a la derecha, a la izquierda, arriba o abajo de otro elemento, o centrar un elemento en relación con su padre.

```

2  <RelativeLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      tools:context=".RelativeLayout">
9
10     <ImageView...>
18
19     <View...>
26
27     <TextView
28         android:layout_width="wrap_content"
29         android:layout_height="wrap_content"
30         android:layout_marginTop="20dp"
31         android:padding="5dp"
32         android:text="EditText:"
33         android:layout_below="@+id/vSuperior"
34         android:layout_toLeftOf="@+id/etNombre"/>
35
36     <EditText...>
44
50     <Button...>
56
63     <View...>
64 </RelativeLayout>

```

ConstraintLayout

Permite crear interfaces de usuario flexibles y dinámicas mediante la disposición de vistas en relación entre sí o con el contenedor padre. Mucho más potente y flexible que RelativeLayout o LinearLayout.

Ventajas de Usar ConstraintLayout

- **Rendimiento Mejorado:** Como puedes reducir la cantidad de layouts anidados, la jerarquía de vistas se simplifica, mejorando el rendimiento.
- **Flexibilidad en el Diseño:** Permite crear interfaces de usuario altamente personalizadas y complejas sin necesidad de múltiples *layouts* anidados.
- **Facilidad de Mantenimiento:** Una jerarquía de vistas más simple hace que el código sea más fácil de entender y mantener.
- **Responsividad:** Facilita la creación de diseños que se adaptan de manera eficiente a diferentes tamaños y orientaciones de pantalla.

```

2  <androidx.constraintlayout.widget.ConstraintLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      tools:context=".ConstraintLayout">
9      <!-- Elementos sobre otros elementos-->
10     <ImageView
11         android:id="@+id/vSmall"
12         android:layout_width="50dp"
13         android:layout_height="50dp"
14         android:layout_marginTop="32dp"
15         android:src="@drawable/ic_user"
16         android:translationZ="1dp"
17         app:layout_constraintEnd_toEndOf="parent"
18         app:layout_constraintHorizontal_bias="0.3"
19         app:layout_constraintStart_toStartOf="parent"
20         app:layout_constraintTop_toTopOf="parent" />
21
22     <Button...>
23
24     <androidx.constraintlayout.widget.Guideline...>
25
26     <View...>
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49 </androidx.constraintlayout.widget.ConstraintLayout>

```


TableLayout

Es un tipo de vista que organiza sus hijos en una cuadrícula de filas y columnas, similar a una tabla HTML. Cada fila dentro de un TableLayout es una instancia de la clase TableRow, y dentro de estas filas se pueden agregar otras vistas, como botones, textos, imágenes, etc.

Cantidad Articulo Precio unitario Total			
2	Mouse	35000	70000
1	Teclado	70000	70000
Subtotal			140000
IVA (19 %)			26600
Total:			166600

```
<TableLayout
    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:stretchColumns="3"
    android:padding="16dp"
    tools:context=".TableLayout">

    <TableRow
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/gris">

        <TextView
            android:layout_column="0"
            android:padding="4dp"
            android:text="Cantidad"
            android:textSize="18sp"
            android:textColor="@color/white"/>

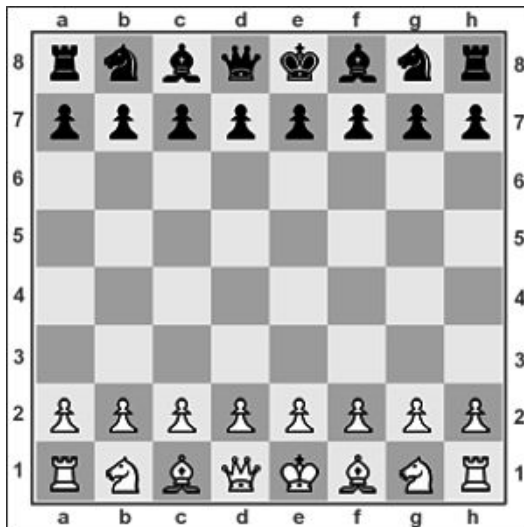
        <TextView...>
        <TextView...>
    </TableRow>

    <TableRow...>

</TableLayout>
```

GridLayout

Organiza sus elementos hijos en una cuadrícula (grid) con filas y columnas. Este tipo de layout es útil cuando necesitas disponer elementos en una estructura de rejilla o tablero.



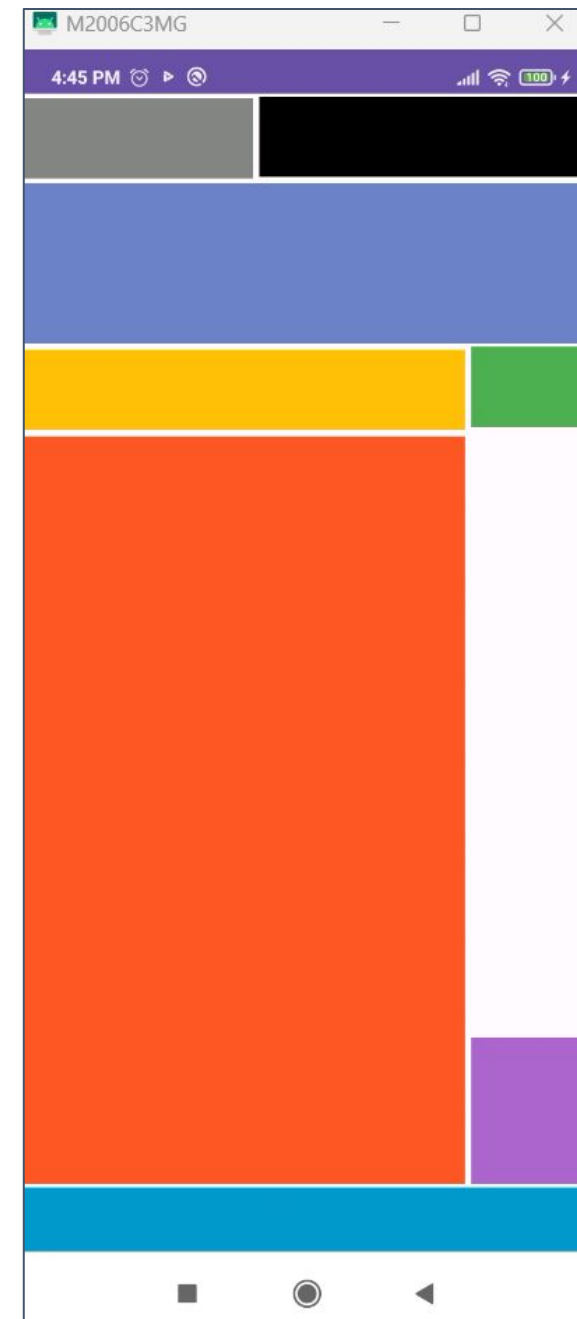
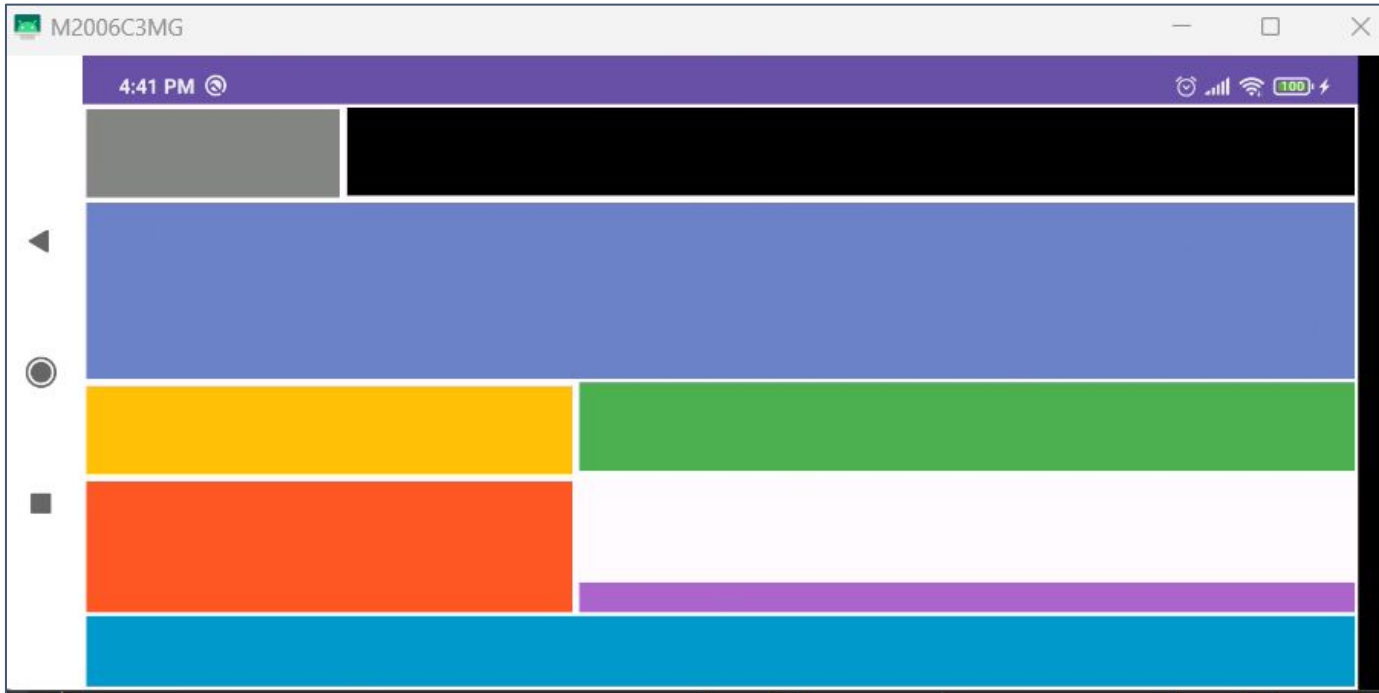
```

2  <GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="wrap_content"
7      android:columnCount="4"
8      android:rowCount="3"
9      tools:context=".GridLayout">
10
11     <View
12         android:layout_width="100dp"
13         android:layout_height="100dp"
14         android:background="@color/black"
15         android:layout_column="0"
16         android:layout_row="0" />
17
18     <View...>
24     <View...>
30
31     <View...>
37
38 </GridLayout>

```

Ejercicio en clase

Realizar el siguiente diseño, únicamente usando ConstraintLayout, tenga en cuenta que al girar la pantalla del celular se debe adaptar los componentes.



GRACIAS !!

*“El esfuerzo vence al talento cuando el talento
no se está esforzando”.*
Anónimo



Bibliografía:

- ❖ <https://developer.android.com/docs?hl=es-419>

