INTERFACE DE USUARIO: LAYOUTS

Programación multimedia y dispositivos móviles 2º DAM

ÍNDICE DE CONTENIDOS

- 1. Interface de usuario. Clases asociadas
- 2. Componentes y layouts
- 3. Linear Layout
- 4. Relative Layout
- 5. Table Layout
- 6. Grid Layout
- 7. Absolute Layout
- 8. Frame Layout
- 9. Constraint Layout
- 10. Scroll View
- 11. Horizontal Scroll View

El interfaz de una aplicación Android es aquello que aparece en pantalla, que por tanto el usuario puede ver, y por tanto, interactuar con él. Todos los componentes que nos ofrece Android derivan de la clase View, y podremos definirlo en el archivo XML del directorio res/layout.

La clase View tiene una serie de atributos que serán heredados por todos los elementos que de ella provienen.

https://developer.android.com/reference/android/view/View

Hay dos atributos obligatorios para cualquier componente/layout:

- layout_width
- layout_height

La clase View tiene una serie de atributos que serán heredados por todos los elementos que de ella provienen.

https://developer.android.com/reference/android/view/View

Hay dos atributos obligatorios para cualquier componente/layout:

- layout_width
- layout_height

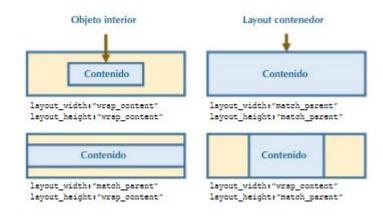
Estos atributos establecen el tamaño que ocupan en la pantalla.

Formas de establecer el tamaño:

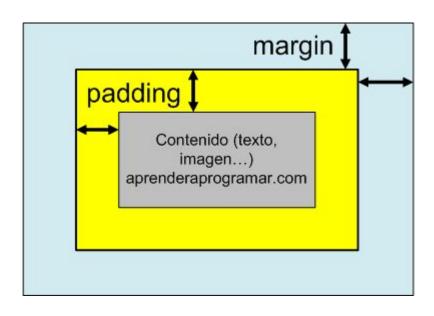
- En pulgadas (in)
- En milímetros (mm)
- En puntos (pt)
- En píxeles (px)
- En píxeles independientes de la densidad (dp)
- En píxeles independientes de la escala (sp)

Otra forma de indicarlo es con:

- match_parent
- wrap_content



Atributos de posicionamiento	layout_width	ancho
	layout_height	alto
Atributos para los márgenes	layout_margin	cuatro márgenes
	layout_marginBotton	margen inferior
	layout_marginLeft	margen izquierdo
	layout_marginRight	margen derecho
	layout_marginTop	margen superior
Atributos para el espaciado	android:padding	espaciado a los cuatro lados
	android:paddingTop	espaciado superior
	android:paddingBottom	espaciado inferior
	android:paddingLeft	espaciado izquierdo
	android:paddingRight	espaciado derecho



Otro atributo muy importante (no obligatorio) es el identificador.

android:id="@+id/texto"

De esta forma, establecemos un nombre para cada uno de los elementos gráficos, para después poder utilizarlos.

- A nivel de diseño. Ejemplo: indicar que un elemento va a la derecha de otro.
- A nivel de código. Ejemplo: usar el elemento seleccionado de un spinner.

2. CONTROLES Y LAYOUT.

- Los controles son los elementos que componen la interfaz de usuario de una aplicación. Entre ellos, se encuentran los botones.
- Un layout es un conjunto de vistas agrupadas de una determinada forma. Los layouts solo definen la disposición de los elementos, no son visibles en la actividad.

Tanto los controles como los layout heredan de la clase View, por lo que se pueden definir tanto desde código Java, como desde XML.

3. LINEAR LAYOUT

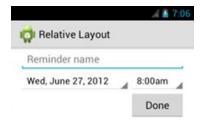
LinearLayout es un grupo de vista que alinea todos los campos secundarios en una única dirección, de manera vertical u horizontal. Puedes especificar la dirección del diseño con el atributo android:orientation.



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"</pre>
    android: layout width="match parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android: paddingRight="16dp"
    android:orientation="vertical" >
    <EditText
        android:layout width="match parent"
        android:layout height="wrap content"
        android:hint="@string/to" />
    <EditText
        android:layout_width="match_parent"
        android:layout height="wrap content"
        android:hint="@string/subject" />
    <EditText
        android:layout width="match parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:gravity="top"
        android:hint="@string/message" />
    <Button
        android:layout width="100dp"
        android:layout height="wrap content"
        android:layout gravity="right"
        android:text="@string/send" />
</LinearLayout>
```

4. RELATIVE LAYOUT

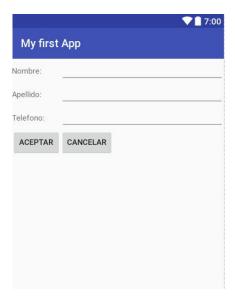
Permite colocar elementos en cualquiera de las cuatro esquinas del contenedor e ir añadiendo otros elementos pegados a estos.



```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"</pre>
    android:layout width="match parent"
    android:layout height="match parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp" >
    <EditText
        android:id="@+id/name"
        android:layout width="match parent"
        android:layout height="wrap content"
        android:hint="@string/reminder" />
    <Spinner
        android:id="@+id/dates"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_below="@id/name"
        android:layout_alignParentLeft="true"
        android:layout_toLeftOf="@+id/times" />
    <Spinner
        android:id="@id/times"
        android:layout width="96dp"
        android:layout height="wrap content"
        android:layout below="@id/name"
        android:layout alignParentRight="true" />
    <Button
        android:layout width="96dp"
        android:layout_height="wrap_content"
        android:layout_below="@id/times"
        android:layout alignParentRight="true"
        android:text="@string/done" />
</RelativeLavout>
```

5. TABLE LAYOUT

Distribuye los elementos de forma tabular, similar a las tablas HTML. Se utiliza la etiqueta <TableRow> para crear una nueva fila.

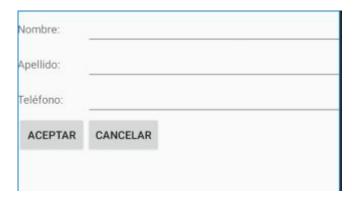


```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout width="match parent"
    android: layout height="match parent">
    <TableRow>
        <TextView
            android:text="Nombre: "/>
       <EditText android:id="@+id/edit message"
            android:layout weight="1"
            android:layout width="0dp"
            android: layout height="wrap content"
            android:hint="" />
    </TableRow>
    <TableRow>
        <TextView
            android:text="Apellido: "/>
        <EditText android:id="@+id/edit message2"
            android:layout weight="1"
            android: layout width="0dp"
            android:layout height="wrap content"
            android:hint="" />
    </TableRow>
    <TableRow>
       <TextView
            android:text="Telefono: "/>
        <EditText android:id="@+id/edit message3"
           android:layout_weight="1"
            android:layout width="0dp"
           android: layout height="wrap content"
            android:hint="" />
    </TableRow>
    <TableRow>
        <Button
            android:text="Aceptar"/>
        <Button
            android:text="Cancelar"/>
    </TableRow>
</TableLayout>
```

6. GRID LAYOUT

Similar a Table Layout, utiliza un interfaz de forma tabular, distribuido en filas y columnas mediante:

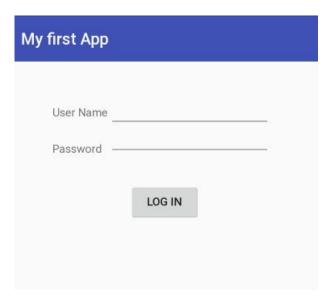
- rowCount
- columnCount



```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"</pre>
    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"
    android:rowCount="4"
    android:columnCount="2">
    <TextView
        android:text="Nombre:"/>
    <TextView
        android:text="Apellido:"/>
    <TextView
        android:text="Teléfono:"/>
    <Button
        android:text="Aceptar"/>
    <EditText
        android:layout height="wrap content"
        android:layout width="match parent"
        android:id="@+id/edit message"
        android:hint=""/>
    <EditText
        android:layout height="wrap content"
        android:layout width="match parent"
        android:id="@+id/edit message2"
        android:hint=""/>
    <EditText
        android:layout height="wrap content"
        android:layout width="match parent"
        android:id="@+id/edit message3"
        android:hint=""/>
    <Button
        android:text="Cancelar"/>
</GridLayout>
```

7. ABSOLUTE LAYOUT

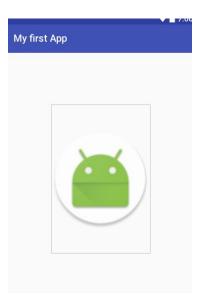
Permite fijar las coordenadas X,Y en las que se visualizará cada elemento de forma arbitraria. Su uso está totalmente no recomendado. (DEPRECATED!!)



```
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"</pre>
    android:layout width="fill parent"
   android: layout height="fill parent">
   <TextView
        android:layout x="110px"
        android: layout v="110px"
        android:text="User Name"
        android:layout width="wrap content"
        android:layout height="wrap content" />
   <EditText
        android:layout x="250px"
        android:layout v="80px"
        android:width="100px"
        android: layout width="200dp"
        android: layout height="wrap content" />
   <TextView
        android: layout x="110px"
        android:layout v="200px"
        android:text="Password"
       android:layout width="wrap content"
        android:layout height="wrap content" />
   <EditText
        android:layout x="250px"
        android: layout v="150px"
        android:width="100px"
        android:layout width="200dp"
        android:layout height="wrap content" />
        android: layout width="wrap content"
        android:layout height="wrap content"
        android:text="Log In"
        android:layout x="300px"
        android:layout_y="300px"/>
</AbsoluteLayout>
```

8. FRAME LAYOUT

Posiciona las vistas usando todo el contenedor sin distribuirlas especialmente. Se utiliza cuando queremos que dos elementos ocupen un mismo lugar. Mostrando uno y ocultando otro.



```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"</pre>
    android: layout width="fill parent"
    android: layout_height="fill_parent"
    android:id="@+id/framelayout" >
    <TextView
        android:id="@+id/frameText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Imagen de muestra"
        android:textSize="15sp"
        android:textStyle="bold"
        android:visibility="gone"
        android:layout gravity="center" />
    <ImageView
        android:id="@+id/frameImage"
        android:layout width="200dp"
        android:layout_height="300dp"
        android: layout_gravity="center"
        android:clickable="true"
        android:src="@mipmap/ic launcher round" />
</FrameLavout>
```

9. CONSTRAINT LAYOUT

Constraint Layout permite colocar los widgets de una forma más flexible, y también redimensionarlos.

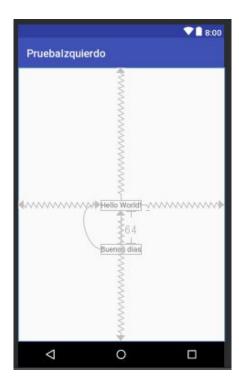
Cuando en una actividad tenemos un gran anidamiento de Layouts, ralentiza mucho la aplicación, e incluso, puede que algunos dispositivos no consigan mostrar la interfaz.

There are currently various types of constraints that you can use:

- · Relative positioning
- Margins
- · Centering positioning
- · Circular positioning
- · Visibility behavior
- · Dimension constraints
- Chains
- · Virtual Helpers objects
- Optimizer

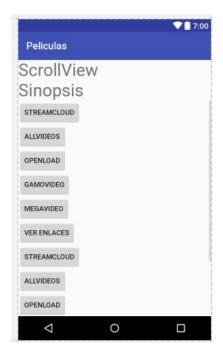
9. CONSTRAINT LAYOUT

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"</p>
   xmlns:tools="http://schemas.android.com/tools"
   android: layout width="match parent"
   android:layout height="match parent"
   <TextView
       android:id="@+id/textView"
       android:layout width="wrap content"
       android: layout height="wrap content"
       android:text="Hello World!"
       app:layout constraintBottom toBottomOf="parent"
       app:layout constraintLeft toLeftOf="parent"
       app:layout constraintRight toRightOf="parent"
       app:layout constraintTop toTopOf="parent" />
   <TextView
       android:id="@+id/textView2"
       android:layout width="wrap content"
       android:layout height="wrap content"
       android:layout marginTop="64dp"
       app:layout constraintStart toStartOf="@+id/textView"
       app:layout_constraintTop_toBottomOf="@+id/textView"
       app:layout goneMarginLeft="23dp" />
</android.support.constraint.ConstraintLayout>
```



10. SCROLL VIEW

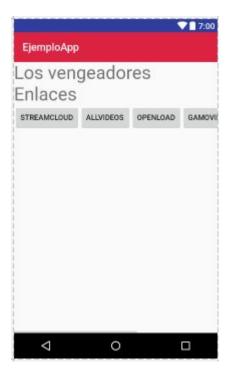
Visualiza una columna de elementos; cuando estos no caben en pantalla se permite un deslizamiento vertical.



```
<ScrollView
    android:layout_width="match_parent"
    android: layout height="match parent">
<LinearLayout
    android:layout_width="match_parent"
    android:layout height="wrap content"
    android:orientation="vertical" >
    <Button
        android:layout width="wrap content"
        android:layout height="wrap content"
        android:text="StreamCloud"/>
    <Button
        android:layout width="wrap content"
        android:layout height="wrap content"
        android:text="AllVideos"/>
    <Button
        android:layout width="wrap content"
        android:layout height="wrap content"
        android:text="OpenLoad"/>
    <Button
        android:layout width="wrap content"
        android:layout height="wrap content"
        android:text="GamoVideo"/>
    <Button
        android:layout width="wrap content"
        android: layout height="wrap content"
        android:text="MegaVideo"/>
```

11. HORIZONTAL SCROLL VIEW

Visualiza una fila de elementos; cuando estos no caben en pantalla se permite un deslizamiento horizontal.



```
<HorizontalScrollView</pre>
   android: layout width="match_parent"
   android: layout height="match parent">
    <LinearLavout
        android:layout width="match parent"
        android: layout height="wrap content"
        android:orientation="horizontal" >
       <Button
           android:layout width="wrap content"
           android:layout_height="wrap_content"
           android:text="StreamCloud"/>
        <Button
            android:layout_width="wrap_content"
            android:layout height="wrap content"
            android:text="AllVideos"/>
        <Button
            android:layout width="wrap content"
            android: layout height="wrap content"
            android: text="OpenLoad"/>
        <Button
            android:layout_width="wrap_content"
            android: layout height="wrap content"
            android: text="GamoVideo"/>
        <Button
            android: layout width="wrap content"
            android: layout_height="wrap_content"
            android: text="MegaVideo"/>
        <Button
            android:id="@+id/btn_act"
            android:layout width="wrap content"
            android: layout height="wrap content"
            android:text="Ver enlaces" >
        </Button>
    </LinearLayout>
</HorizontalScrollView>
```

10. DUDAS Y PREGUNTAS

