cpifp Los Enlaces

PROGRAMACIÓN MULTIMEDIA Y DISPOSITIVOS MÓVILES

2º DESARROLLO DE APLICACIONES MULTIPLATAFORMA

TEMA 1. LAYOUTS

T1. LAYOUTS

0. TIPOS DE LAYOUTS

Un ***layout*** es una distribución de elementos y formas dentro de un diseño. En *Android Studio* contamos con distintos tipos de *layouts*, los cuales se definen mediante código XML. Éstos son:

- FrameLayout
- LinearLayout
- TableLayout
- ConstraintLayout
- RelativeLayout (no se usa)
- AbsoluteLayout (no se usa)

cpi fp Los Enlaces

T1. LAYOUTS

0. TIPOS DE LAYOUTS

Al crear un nuevo proyecto en *AndroidStudio* se crea por defecto un archivo *layout_activity.xml* en la ruta ***app*** → ***res*** → ***layout*** de la vista *Android* en la pestaña *Project* con el siguiente contenido:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    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"
    tools:context=".firstapp.ConstraintLayout">

</androidx.constraintlayout.widget.ConstraintLayout>
```

0. TIPOS DE LAYOUTS

Los atributos *layout_width* y *layout_height* indican el espacio a ocupar por el contenido. El valor *match_parent* ajusta el *layout* al contenedor padre de este elemento, es decir, a la pantalla del dispositivo.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    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"
    tools:context=".firstapp.ConstraintLayout">

</androidx.constraintlayout.widget.ConstraintLayout>
```

cpifp Los Enlaces

T1. LAYOUTS

1. FRAMELAYOUT

En un *FrameLayout* los elementos contenidos en él se alinean al centro, arriba, abajo, a izquierda (*start*) y derecha (*end*) de la pantalla mediante la propiedad *layout_gravity*.

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".firstapp.FrameLayout">

</FrameLayout>
```

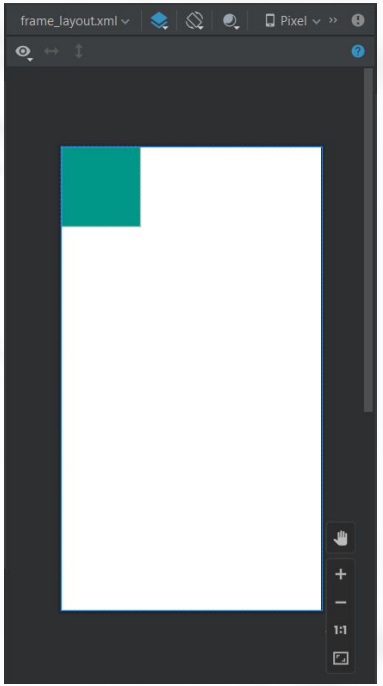
cpifp Los Enlaces

T1. LAYOUTS

1. FRAMELAYOUT

Una *View* es un tipo de elemento que el usuario puede ver y con el que puede interactuar. Dentro de un *FrameLayout* se podría definir una *View* con:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout.../>
<View
   android:layout_width="125dp"
   android:layout_height="125dp"
   android:background="#009688"
   android:layout_gravity="top|start"/>

</FrameLayout>
```



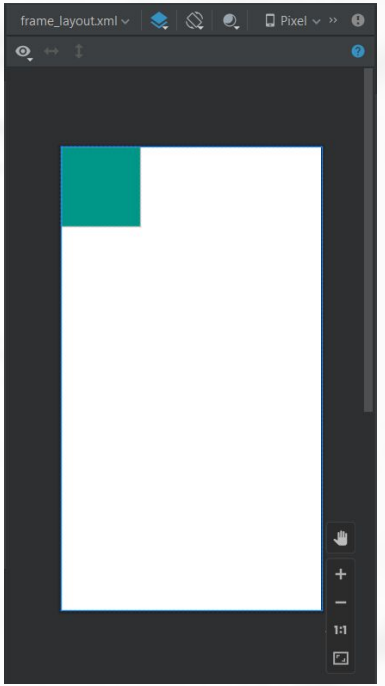
cpifp Los Enlaces

T1. LAYOUTS

1. FRAMELAYOUT

El atributo *background* determina el color de fondo de la *View* y el atributo *layout_gravity* indica la posición donde se coloca ésta.

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout.../>
<View
   android:layout_width="125dp"
   android:layout_height="125dp"
   android:background="#009688"
   android:layout_gravity="top|start"/>

</FrameLayout>
```

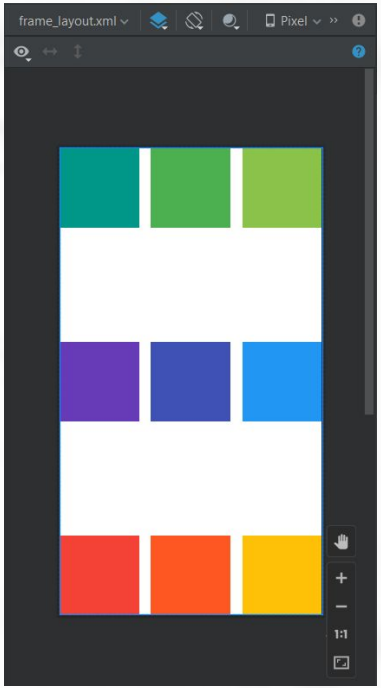


cpi'fp Los Enlaces

T1. LAYOUTS

1. FRAMELAYOUT

Intenta dibujar el siguiente diseño con *FrameLayout* y las propiedades *background* y *layout_gravity*:



- Crea 9 elementos *view* de la misma anchura y altura y sitúalos en las posiciones que se ven en la figura.
- Cámbiales el color mediante la ventana de colores predeterminados que se despliega al pinchar sobre el color a la izquierda del código de cada <View/>.
- Elige la pestaña *Custom* y el grupo *Material 600*.

2. LINEARLAYOUT

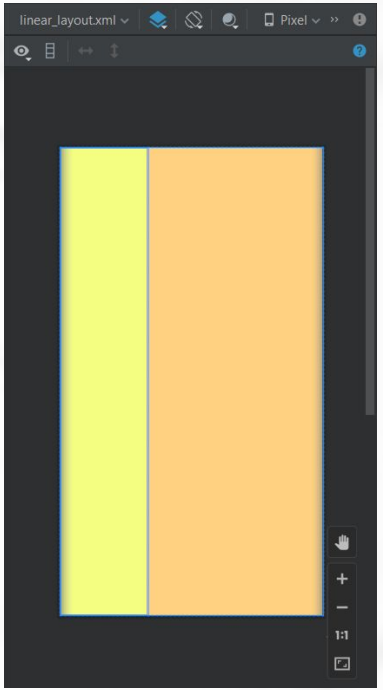
En un *LinearLayout* los elementos contenidos en él se alinean horizontal o verticalmente, según se defina en el atributo *orientation*.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".firstapp.LinearLayout"
   android:orientation="horizontal">

</LinearLayout>
```

2. LINEARLAYOUT

Se puede asignar un peso (*layout_weight*) a cada elemento para que ocupen un espacio relativo:

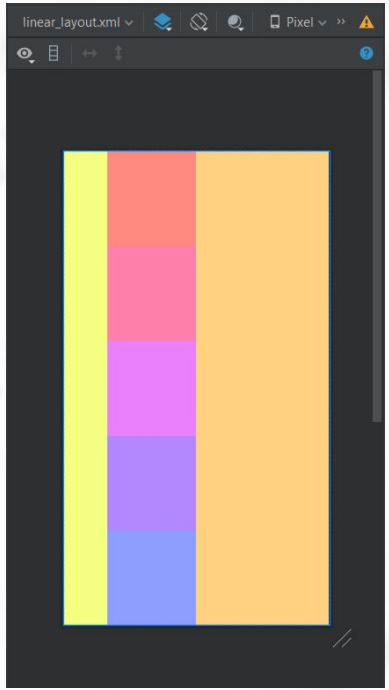
```
<LinearLayout...
   <View
       android:layout_width="0dp"
       android:layout_height="match_parent"
       android:layout_weight="1"
       android:background="#F4FF81"/>
   <View
      android:layout_width="0dp"
      android:layout_height="match_parent"
      android:layout_weight="2"
      android:background="#FFD180"/>
</LinearLayout>
```



2. LINEARLAYOUT

También se pueden anidar distintos *LinearLayout* para crear diversas formas.

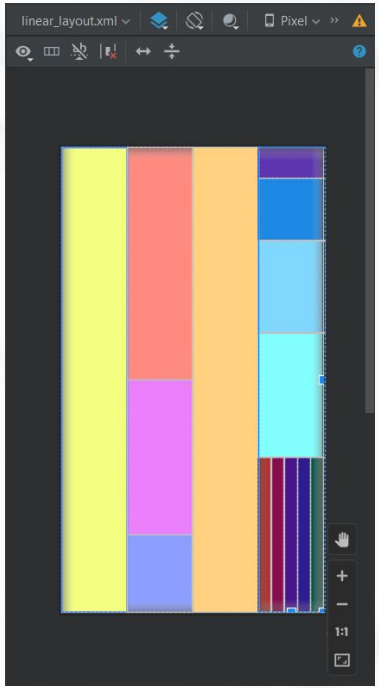
```
<View
   android:layout_width="0dp"
   android:layout_height="match_parent"
   android:layout_weight="1"
   android:background="#F4FF81"/>
<LinearLayout
   android:layout_width="0dp"
   android:layout_height="match_parent"
   android:layout_weight="2"
   android:orientation="vertical">
   <View...>
   <View...>
```



2. LINEARLAYOUT

Intenta dibujar el siguiente diseño con *LinearLayout* anidados y las propiedades *orientation* y *layout_weight*:

- La segunda columna tiene 3 elementos *View* con pesos 3, 2 y 1 respectivamente.
- La cuarta columna tiene 5 elementos *View* con pesos 1, 2, 3, 4 y 5 respectivamente.
- El peso en anchura de los elementos dentro de los *layouts* horizontales es en todos 1.



3. TABLELAYOUT

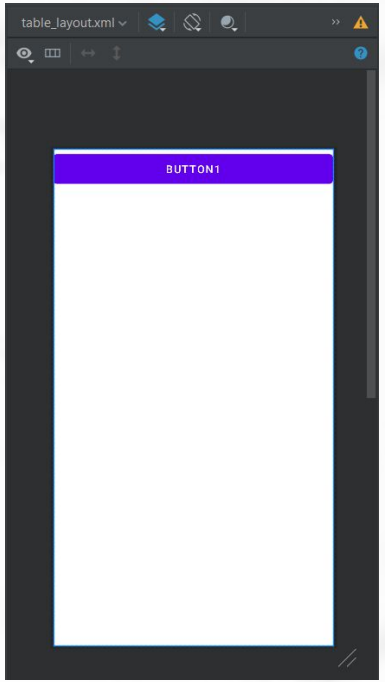
En un *TableLayout* los elementos se distribuyen por filas pudiendo crear tantas filas como queramos. Los elementos de cada fila se distribuyen como un *LinearLayout* con orientación horizontal.

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".firstapp.TableLayout">

</TableLayout>
```

3. TABLELAYOUT

Un *Button* es un tipo de elemento que puede contener texto y con el que se puede interactuar. Dentro de una fila de la tabla se podría insertar con:

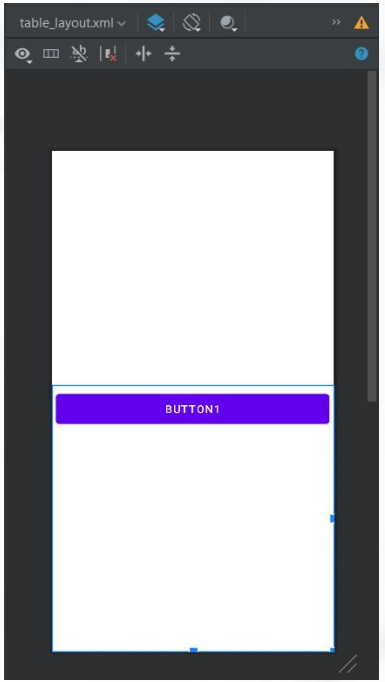
```
<TableRow
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:layout_weight="1">

 <Button
     android:layout_width="0dp"
     android:layout_height="50dp"
     android:layout_weight="1"
     android:text="Button1"/>
```



3. TABLELAYOUT

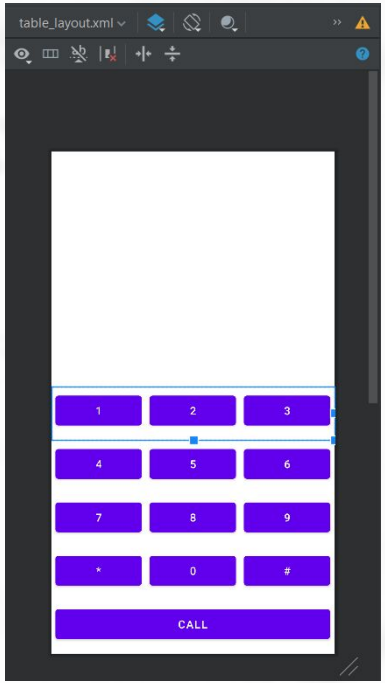
Los márgenes de cualquier elemento se pueden ajustar con las propiedades *layout_margin*, *layout_marginTop*, *layout_marginHorizontal*, *layout_marginStart*,...

```
<TableRow
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:layout_marginTop="300dp"
   android:layout_weight="1">

 <Button
     android:layout_width="0dp"
     android:layout_height="50dp"
     android:layout_weight="1"...
```



3. TABLELAYOUT

Intenta dibujar el siguiente diseño con *TableLayout* utilizando las distintas propiedades de márgenes y pesos para la anchura de los botones:



- Crea 5 filas con 3 elementos en cada una, salvo en la última.
- Ajusta el margen superior de la primera fila para que se separe del borde superior de la pantalla.
- Ponles a todos los botones el mismo margen general.

4. CONSTRAINTLAYOUT

En un *ConstraintLayout* los elementos se distribuyen por posiciones relativas a los márgenes de la pantalla y al resto de elementos.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    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"
    tools:context=".firstapp.ConstraintLayout">

</androidx.constraintlayout.widget.ConstraintLayout>
```

4. CONSTRAINTLAYOUT

Para poder anclar otros elementos a un determinado elemento hay que propocionarle una *id*. Las restricciones se realizan con la propiedad *constraint*.

```
<View
    android:id="@+id/viewBlue"
    android:layout_width="match_parent"
    android:layout_height="125dp"
    android:background="@color/teal_200"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"/>
```

4. CONSTRAINTLAYOUT

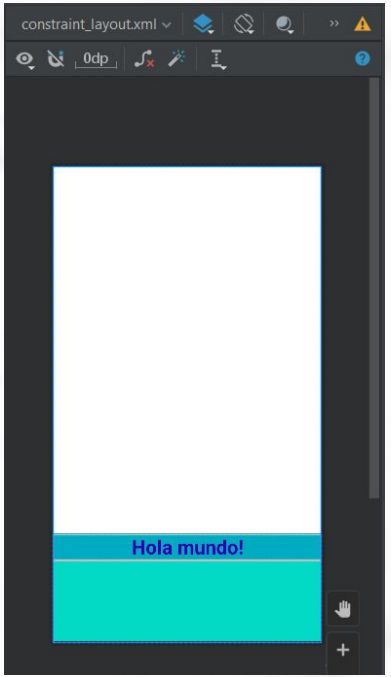
Se puede elegir rápidamente las propiedades *constraint* escribiendo simplemente `botbot`, `toptop`, `ststo` `endend`. *AndroidStudio* mostrará un desplegable con ellas.

```
<View
    android:id="@+id/viewBlue"
    android:layout_width="match_parent"
    android:layout_height="125dp"
    android:background="@color/teal_200"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"/>
```

4. CONSTRAINTLAYOUT

Elemento *TextView*:

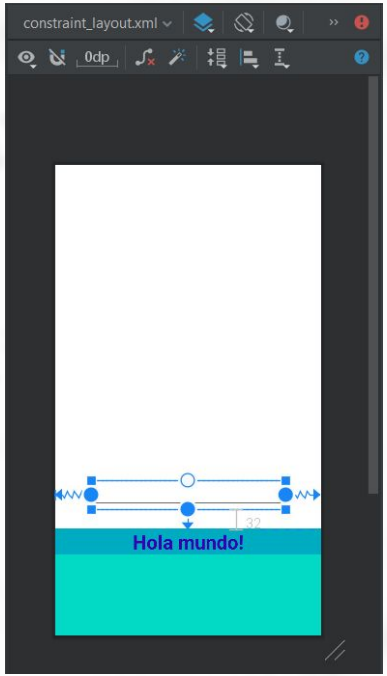
```
<TextView
    android:id="@+id/HolaMundo"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#00ACC1"
    android:text="Hola mundo!"
    android:textAlignment="center"
    android:textColor="@color/purple_700"
    android:textSize="30sp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@id/viewBlue"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent" />
```



4. CONSTRAINTLAYOUT

Elemento *EditText*:

```
<androidx.appcompat.widget.AppCompatEditText
    android:id="@+id/etName"
    android:layout_width="300dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="32dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintBottom_toTopOf="@id/HolaMundo"/>
```



4. CONSTRAINTLAYOUT

Intenta dibujar el siguiente diseño con *ConstraintLayout* utilizando las distintas propiedades de márgenes y restricciones para situar los distintos elementos:

- Los elementos, de arriba abajo, son: *TextView*, *EditText*, *Button*, *EditText* y *View*.
- El texto *Escribe tu nombre* tiene un margen superior de 64dp y el botón y el *EditText* márgenes inferiores de 120dp y 32dp, respectivamente.