Elementos de la interfaz, navegación e interacción

Las pantallas en un proyecto Android

{desafío} latam_

Utilizar elementos de interfaz de usuario básicos del entorno Android para la implementación de un prototipo de acuerdo a las especificaciones entregadas.

- Unidad 1: Ambiente de desarrollo y sus elementos de configuración.
- Unidad 2: Elementos de la interfaz, navegación e interacción.
- Unidad 3: Fundamentos de GIT y GitHub.

Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

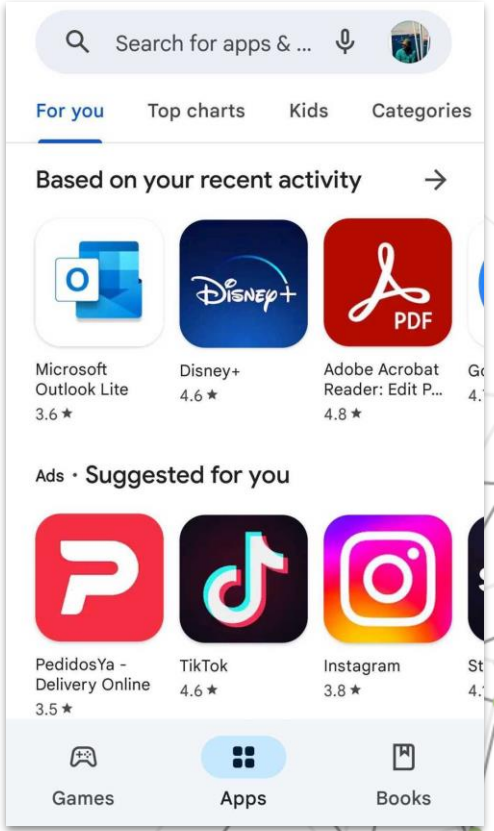
- *Utiliza elementos ViewGroups y Views requeridos para satisfacer la especificación entregada.*

Encontrando Activities y Fragments

{desafío} latam_

Conversemos

- ¿Cuántas actividades tiene potencialmente Google Play Store?
- ¿Cuántos fragmentos se pueden identificar?



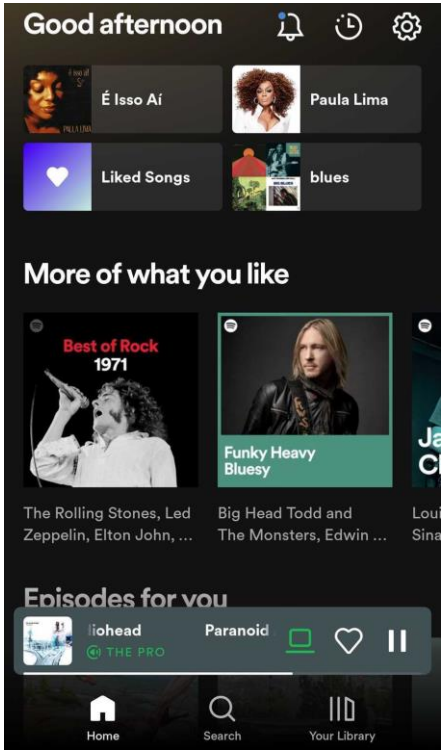
{desafío} latam_

/* Activity & Fragment */

Activity & Fragment

Las actividades son el lugar ideal para colocar elementos globales en la interfaz de usuario de tu app, como por ejemplo, un panel lateral de navegación.

Los fragmentos son más adecuados para definir y administrar la IU de una sola pantalla o de una parte de ella.



/* El editor de vistas */

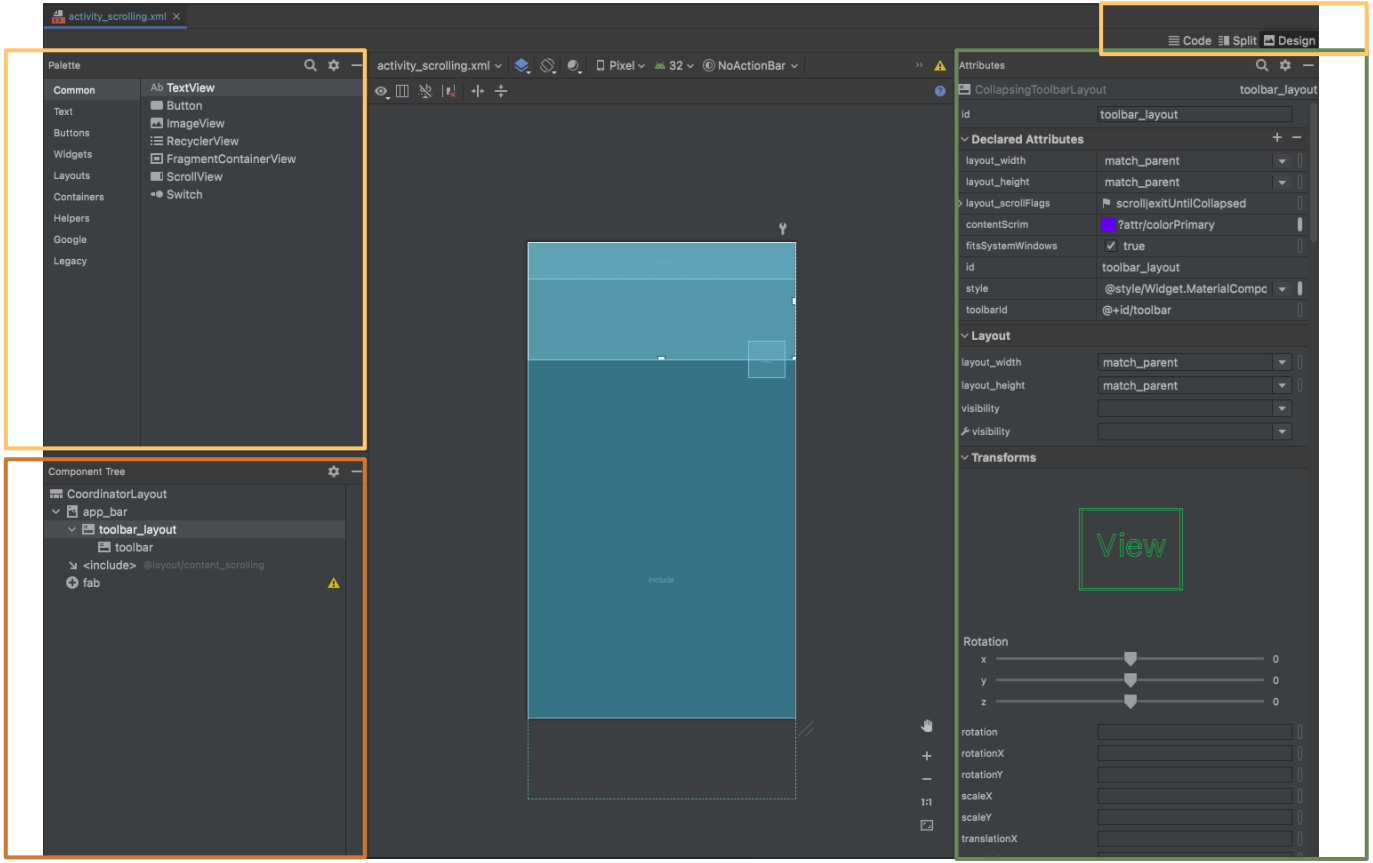
Editor de vistas

- Cada pantalla tiene asociado un **layout** que define la **estructura** de la interfaz de usuario.
- Para modificar los layouts, Android Studio incluye un editor gráfico de vistas que incluye componentes que facilitan el desarrollo de estas interfaces.
- Al abrir un layout, se despliega el editor de vistas con el layout para modificar una paleta de vistas con componentes para agregar imágenes, textos, listas y botones (entre otros), un árbol con la jerarquía de vistas y los atributos asociados a la vista seleccionada.

Editor de vistas

Tipo de vista del editor: código, diseño o ambos

Paleta con widgets disponibles



Atributos de la vista seleccionada

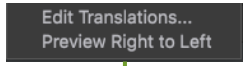
Árbol con la jerarquía de vistas

Desde la raíz a las vistas anidadas

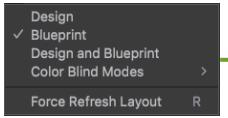
{desafío} latam_

Editor de vistas

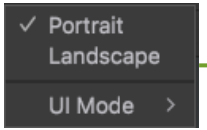
Previsualización de la configuración regional (traducción de textos, RTL)

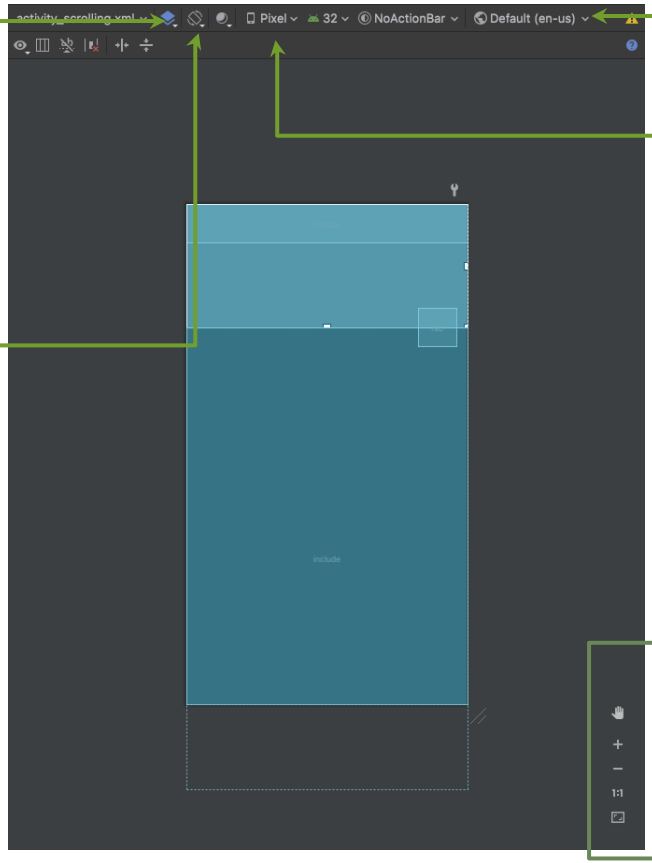


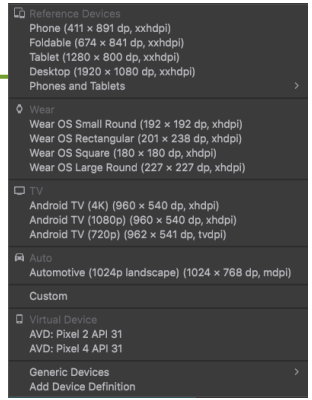
Tipo de superficie de diseño



Orientación de la pantalla







Dispositivo para vista previa. Se puede probar con distintos tamaños de pantalla la vista en diseño

Controles para mover y escalar el tamaño de la pantalla de diseño

{desafío} latam_

/* Views and View Groups */

Views

Vistas

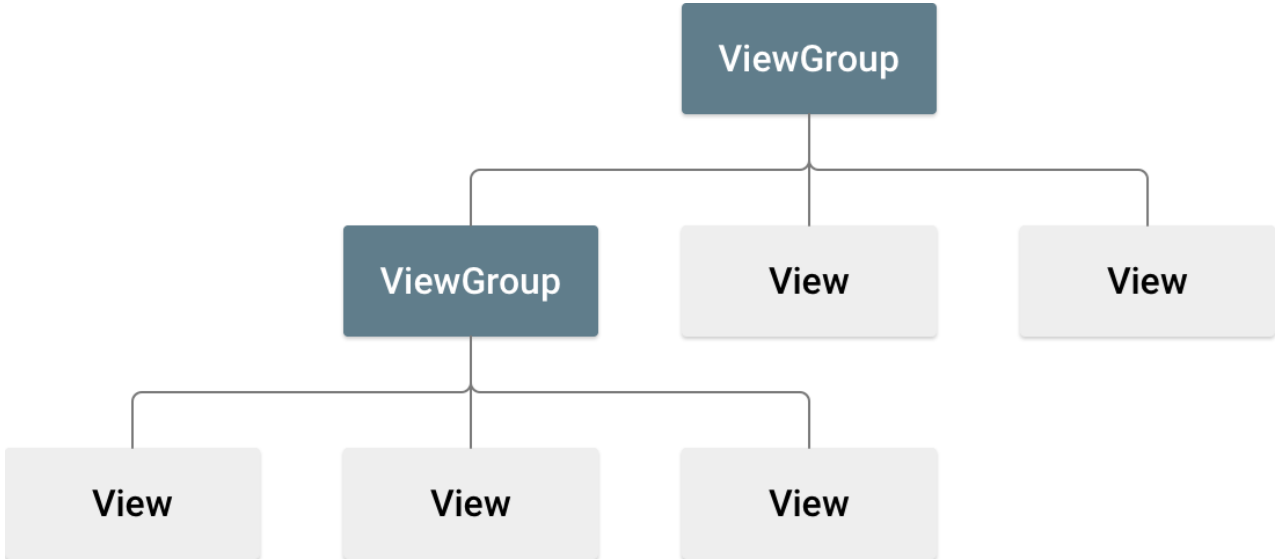
- Todos los elementos (vista o *widget*) en el layout se construyen heredando de View y ViewGroup.
- Un **widget** generalmente dibuja algo que el usuario puede **ver e interactuar**.
- Todo lo que se ve en la pantalla corresponde a vistas independientes.

Layout

Estructura de diseño

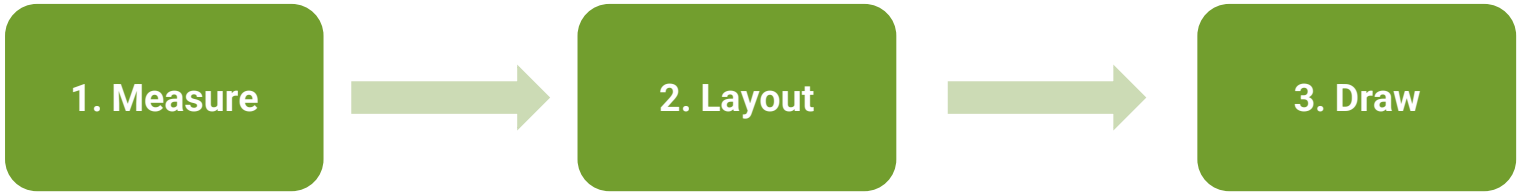
- Cada pantalla tiene asociado un **layout** que define la **estructura** de la interfaz de usuario.
- Un layout es un **ViewGroup** y corresponde a un **contenedor invisible** que define la estructura para un widget y otros ViewGroup.
- La estructura del layout se define en un archivo XML.

View y ViewGroup



Fuente: Diseño.

Pasos para dibujar una pantalla



- **Medición top-down.**
- Se **mide** cada ViewGroup.
- Se mide cada vista.
- Se miden los hijos de cada ViewGroup.

- **Medición top-down**
- Cada ViewGroup determina la **posición** de sus hijos usando los tamaños determinados en la etapa anterior.

- **Medición top-down.**
- Para cada objeto en el view tree se crea un **canvas** para enviar una lista de comandos a la GPU.
- Los comandos incluyen las posiciones y tamaños de Views y ViewGroup.

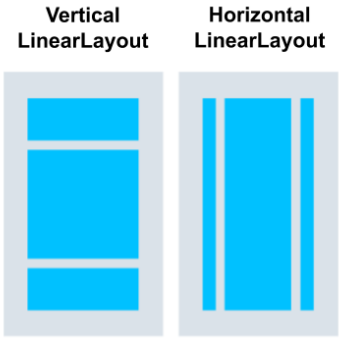
/* Tipos de Layouts */

Tipos de layout

Layout más utilizados

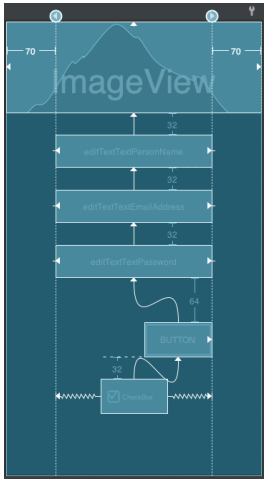
LinearLayout

Los elementos se organizan horizontal y verticalmente.



ConstraintLayout

Permite ordenar los elementos en forma relativa a su contenedor o a otros elementos.



/* Constraint Layout */

ConstraintLayout

Contexto

- Utiliza una jerarquía plana de vistas, lo que hace que sea más rápido en mostrarse, mejorando el rendimiento de la app.
- Es muy flexible y permite ubicar las vistas en forma relativa al contenedor (layout) y a otras vistas.
- Incluye elementos que ayudan a organizar la distribución de las vistas en el diseño.

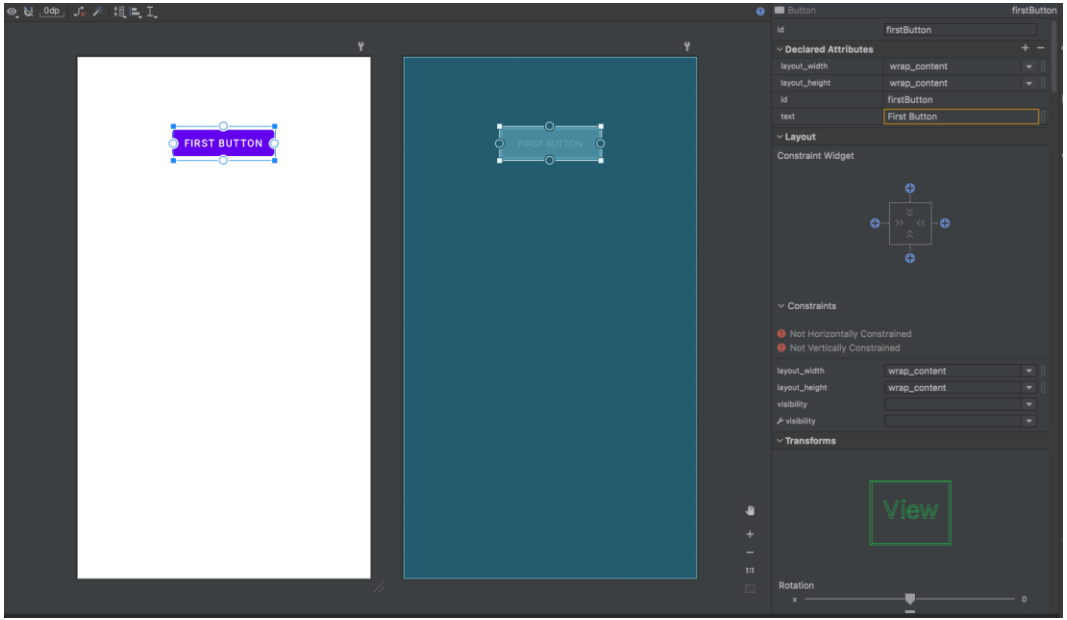
ConstraintLayout

Constraints

Al agregar una nueva vista, se deben proporcionar las restricciones (constraints) indicando dónde se ubicará dentro de la pantalla.

Los 4 círculos en el borde del botón indican las constraints respecto a 4 partes que están presentes en todas las vistas:

- Top (arriba)
- Bottom (abajo)
- Start (comienzo)
- End (fin)



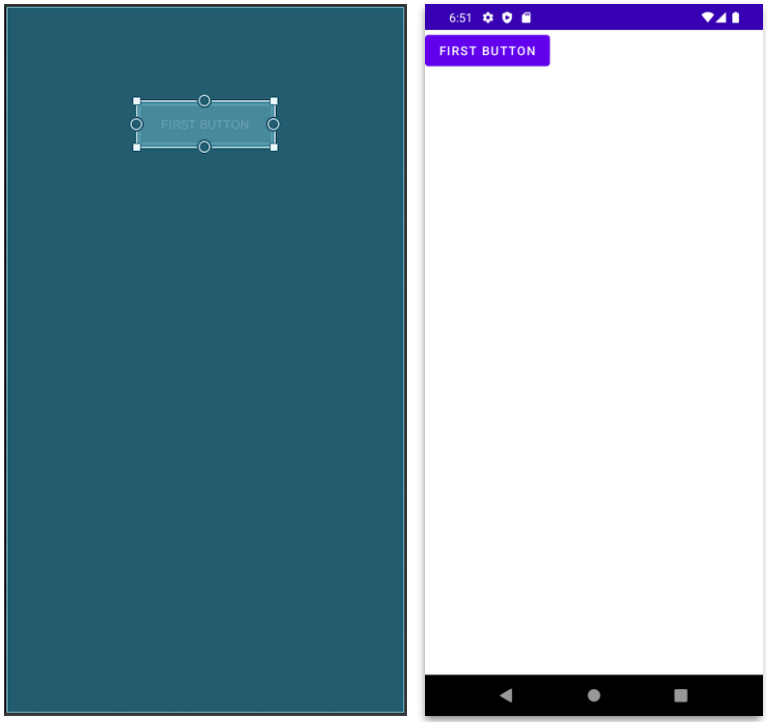
Fuente: Imágenes y códigos ADL.

ConstraintLayout

Constraints ¿y si no agrego las constraints?

- Si la vista no tiene constraints y ejecutamos la app, el botón se ubica en las coordenadas 0,0 de la pantalla.
- A pesar de tener una vista de diseño con el botón centrado, al momento de correr la app no coincide con lo esperado.

¿Por qué “salta” la vista hacia esa posición?



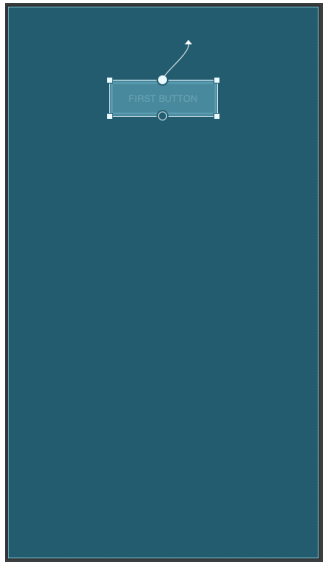
ConstraintLayout

Cómo agregar constraints

Si la vista no tiene constraints marca errores

Se pueden crear desde el panel de Attributes (al costado derecho) con el botón +.

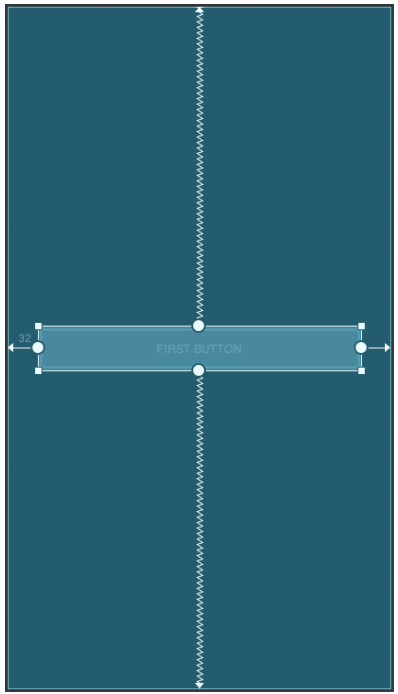
O con el puntero puedes seleccionar uno de los círculos que corresponden a las constraints de la vista y enlazarlo a otra vista. Por ejemplo, a la parte superior de la pantalla dentro del ConstraintLayout



ConstraintLayout

0 dp

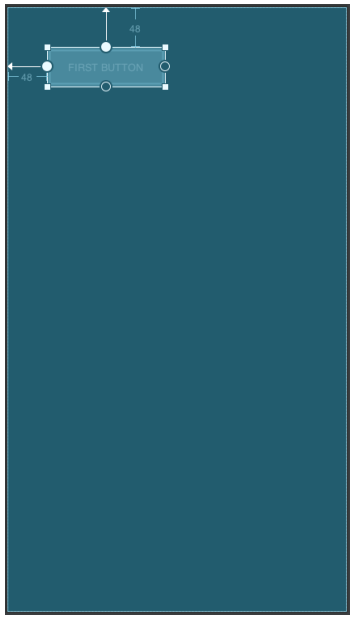
Al indicar el ancho como 0dp, se ocupa todo el espacio disponible respetando los márgenes start y end.




```
<?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:id="@+id/constraintLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/firstButton"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="32dp"
        android:layout_marginEnd="32dp"
        android:text="@string/first_button"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

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

ConstraintLayout

Parent position




```
<?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:id="@+id/constraintLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/firstButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="48dp"
        android:layout_marginTop="48dp"
        android:text="@string/first_button"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

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

Es la posición relativa al layout que contiene a nuestra vista.

Se indican los márgenes start y top para ubicar a firstButton en la parte superior izquierda.

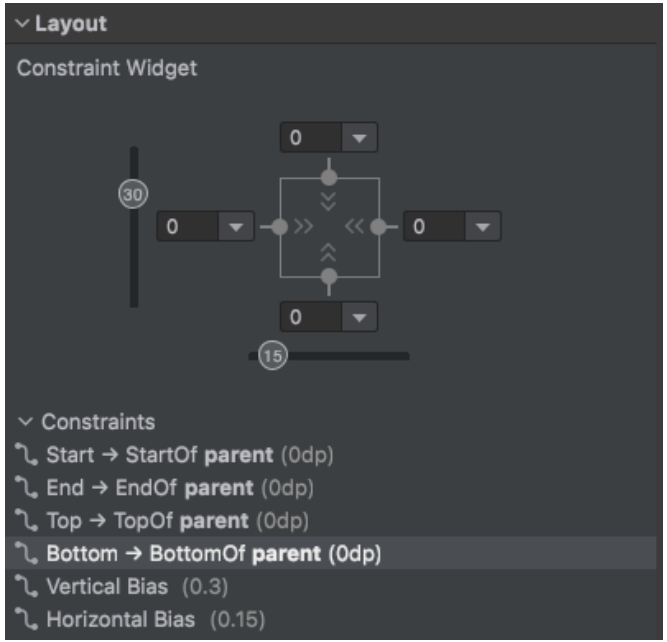
Constraint Layout

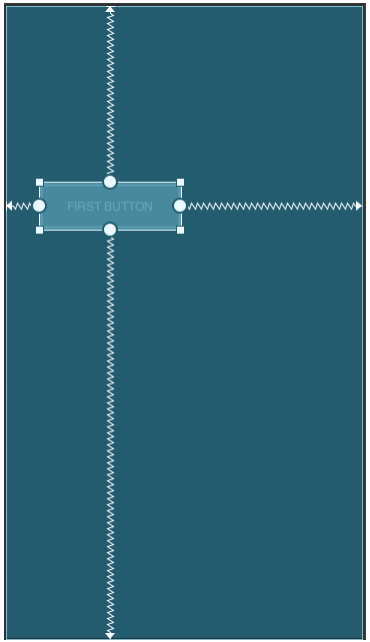
Constraint bias

Permiten indicar la ubicación relativa de la vista.

Indicando un bias de 30 en forma vertical, significa que la vista se va a ubicar en el 30% de la pantalla desde arriba a abajo, adaptándose al tamaño disponible.

Las bias pueden ser verticales y horizontales.



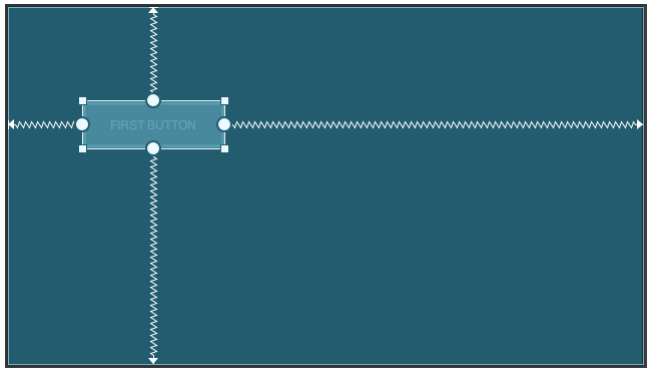


{desafío} latam_

Constraint Layout

Constraint bias

Al visualizar en landscape, el botón sigue manteniendo una ubicación relativa dado el tamaño de la pantalla.



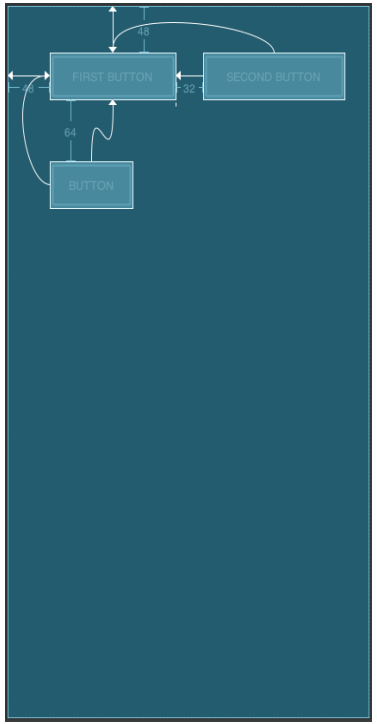

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/constraintLayout"
   android:layout_width="match_parent"
   android:layout_height="match_parent">

   <Button
       android:id="@+id/firstButton"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:text="@string/first_button"
       app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintHorizontal_bias="0.15"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toTopOf="parent"
      app:layout_constraintVertical_bias="0.3" />

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

Constraint Layout

Order position

Si first Button se mueve, los otros botones también se mueven.

Se debe definir al menos una constraint horizontal (start, end) y al menos una vertical (top, bottom).




```xml
<?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:id="@+id/constraintLayout"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <Button
       android:id="@+id/firstButton"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_marginStart="48dp"
       android:layout_marginTop="48dp"
       android:text="@string/first_button"
       app:layout_constraintStart_toStartOf="parent"
       app:layout_constraintTop_toTopOf="parent" />
   <Button
       android:id="@+id/secondButton"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_marginStart="32dp"
       android:text="@string/second_button"
       app:layout_constraintStart_toEndOf="@+id/firstButton"
       app:layout_constraintTop_toTopOf="@+id/firstButton" />
   <Button
       android:id="@+id/thirdButton"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_marginTop="64dp"
       android:text="@string/third_button"
       app:layout_constraintStart_toStartOf="@+id/firstButton"
       app:layout_constraintTop_toBottomOf="@+id/firstButton"
/>
</androidx.constraintlayout.widget.ConstraintLayout>
```

{desafío} latam_

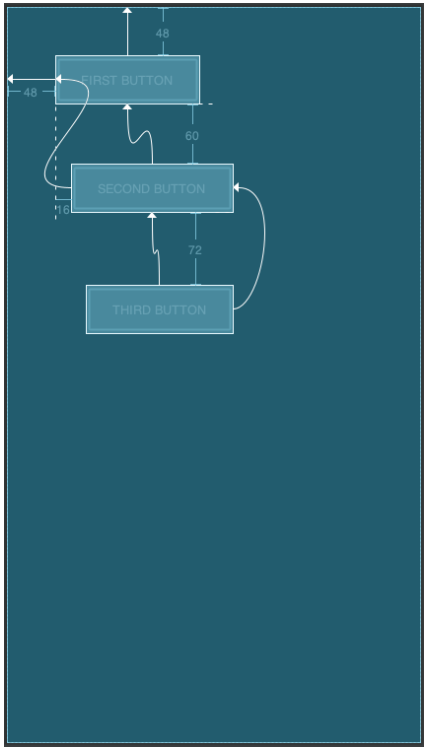
Constraint Layout

Alignment

La constraint start de Second Button apunta donde comienza FirstButton y le agrega un margen de 16 dp para desplazarlo.

La constraint end de ThirdButton apunta donde termina SecondButton.

Si se mueve FirstButton, se moverá SecondButton y como consecuencia ThirdButton.



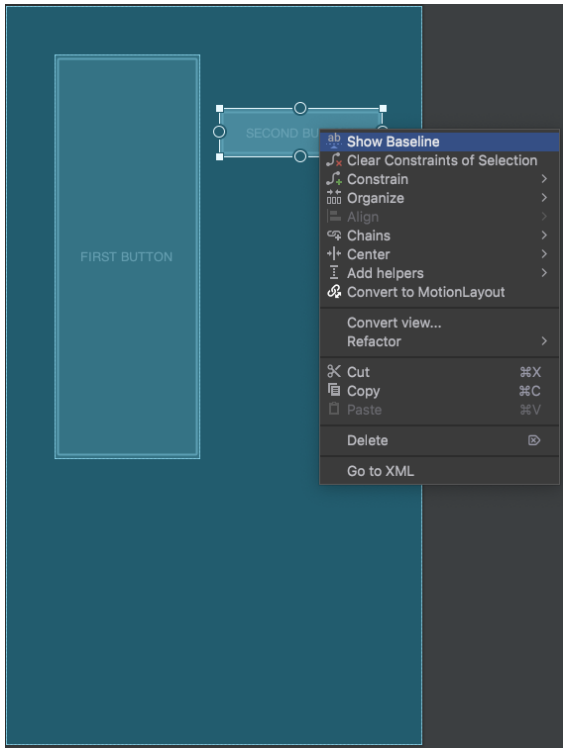

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/constraintLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:id="@+id/firstButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="48dp"
        android:layout_marginTop="48dp"
        android:text="@string/first_button"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <Button
        android:id="@+id/secondButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="60dp"
        android:text="@string/second_button"
        app:layout_constraintStart_toStartOf="@+id/firstButton"
        app:layout_constraintTop_toBottomOf="@+id/firstButton" />
    <Button
        android:id="@+id/thirdButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="72dp"
        android:text="@string/third_button"
        app:layout_constraintEnd_toEndOf="@+id/secondButton"
        app:layout_constraintTop_toBottomOf="@+id/secondButton" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

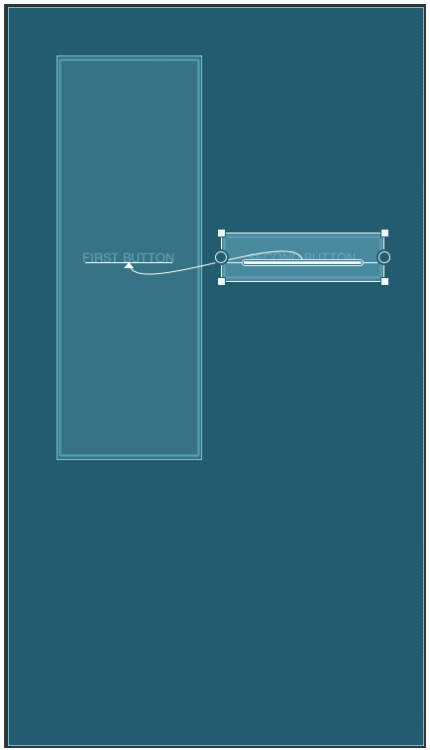
{desafío} latam_

Constraint Layout

Baseline Alignment

El texto de SecondButton se alinea con el texto de First Button independiente de su tamaño.





{desafío} latam_

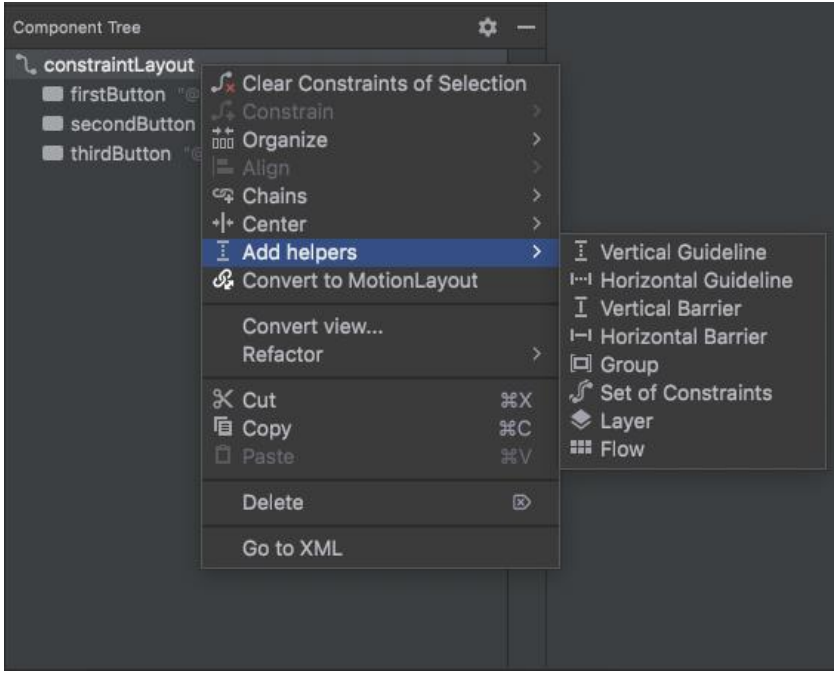
Constraint Layout

Helpers

Son objetos especiales que ayudan al diseño del layout.

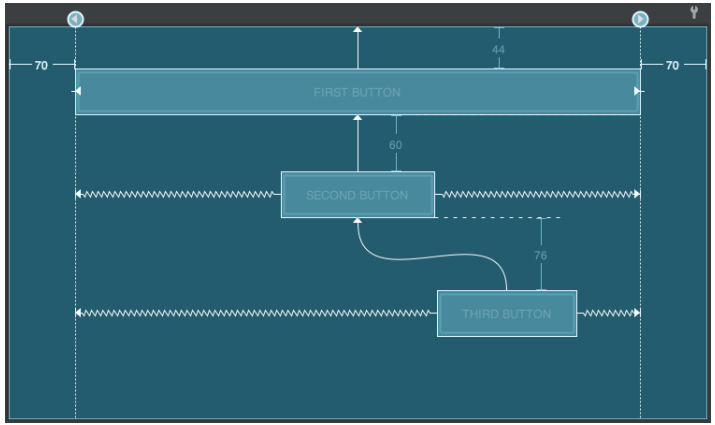
Estos elementos no se muestran en la pantalla del dispositivo y solo sirven para diseñar las pantallas.

Veamos Guideline a modo de ejemplo:



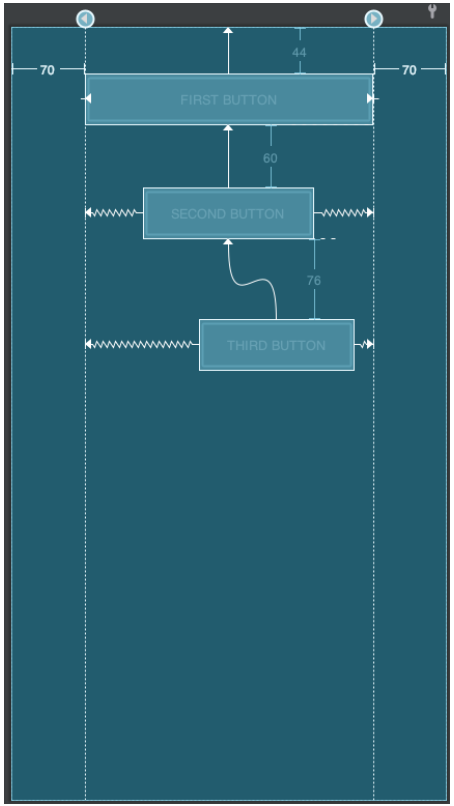
Constraint Layout

Helpers - Guideline



Al agregar una guideline al comienzo y al final de la pantalla permite que todas las vistas respeten el mismo margen.

Si se modifica el margen de la guideline, se mueven todas las vistas que dependen de ella.



{desafío} latam_

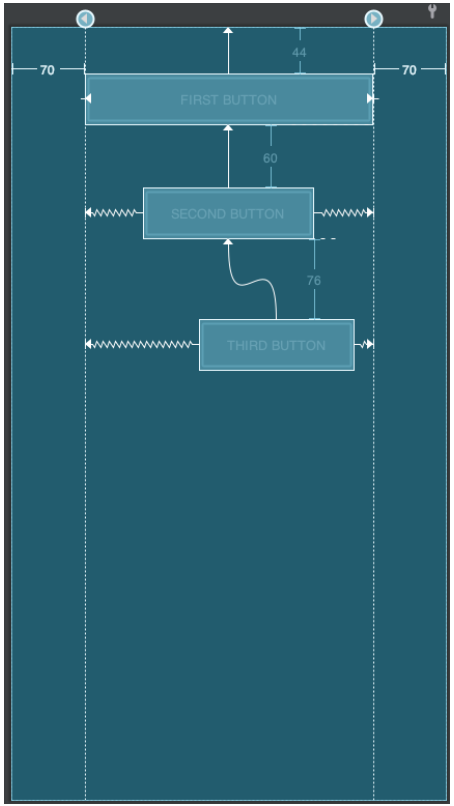
Constraint Layout

Helpers - Guideline

FirstButton ocupa todo el espacio disponible (width=0dp).

SecondButton se ubica al centro horizontalmente entre ambas guidelines.

ThirdButton está usando una bias de 85% relativo al espacio disponible entre ambas guidelines.

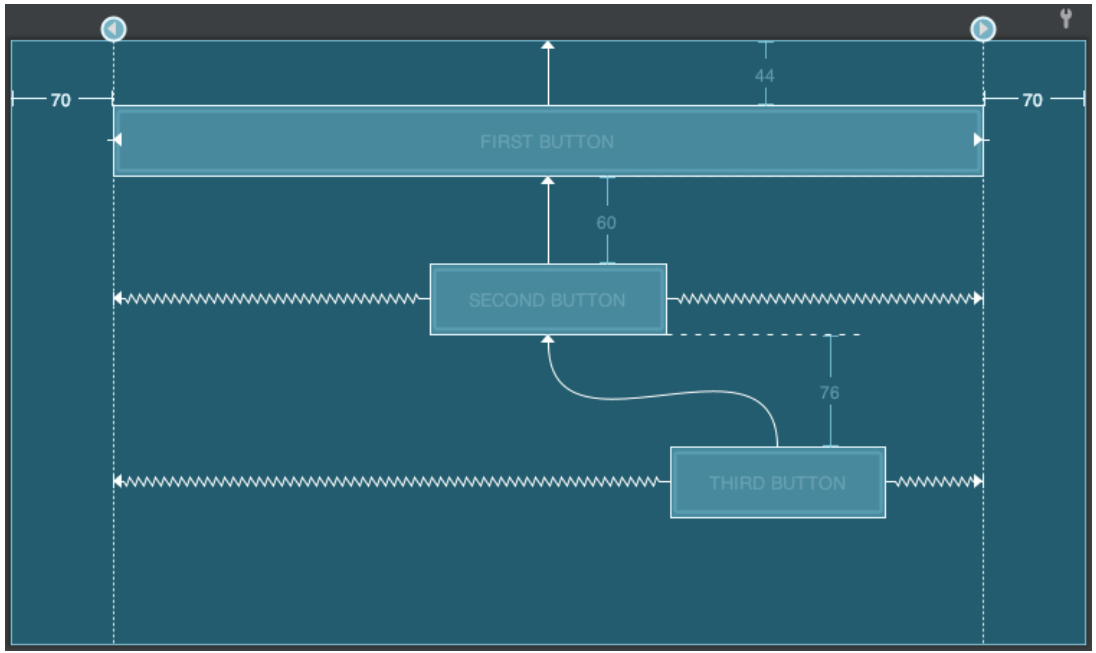



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/constraintLayout"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <Button
       android:id="@+id/firstButton"
       android:layout_width="0dp"
       android:layout_height="wrap_content"
       android:layout_marginTop="44dp"
       android:text="@string/first_button"
app:layout_constraintEnd_toStartOf="@+id/guidelineEnd"
app:layout_constraintStart_toStartOf="@+id/guidelineStart"
       app:layout_constraintTop_toTopOf="parent" />
   <Button
       android:id="@+id/secondButton"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_marginTop="60dp"
       android:text="@string/second_button"
app:layout_constraintEnd_toStartOf="@+id/guidelineEnd"
app:layout_constraintStart_toStartOf="@+id/guidelineStart"
app:layout_constraintTop_toBottomOf="@+id/firstButton" />
   <Button
       android:id="@+id/thirdButton"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_marginTop="76dp"
       android:text="@string/third_button"
    app:layout_constraintEnd_toStartOf="@+id/guidelineEnd"
       app:layout_constraintHorizontal_bias="0.85"
app:layout_constraintStart_toStartOf="@+id/guidelineStart"
app:layout_constraintTop_toBottomOf="@+id/secondButton" />
   <androidx.constraintlayout.widget.Guideline
       android:id="@+id/guidelineStart"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:orientation="vertical"
       app:layout_constraintGuide_begin="70dp" />
   <androidx.constraintlayout.widget.Guideline
       android:id="@+id/guidelineEnd"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:orientation="vertical"
       app:layout_constraintGuide_end="70dp" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

{desafío} latam_

Constraint Layout

Helpers - Guideline



Al cambiar la orientación del layout a landscape, se mantienen las ubicaciones dependiendo del espacio disponible.

Ejercicio - Constraint Layout

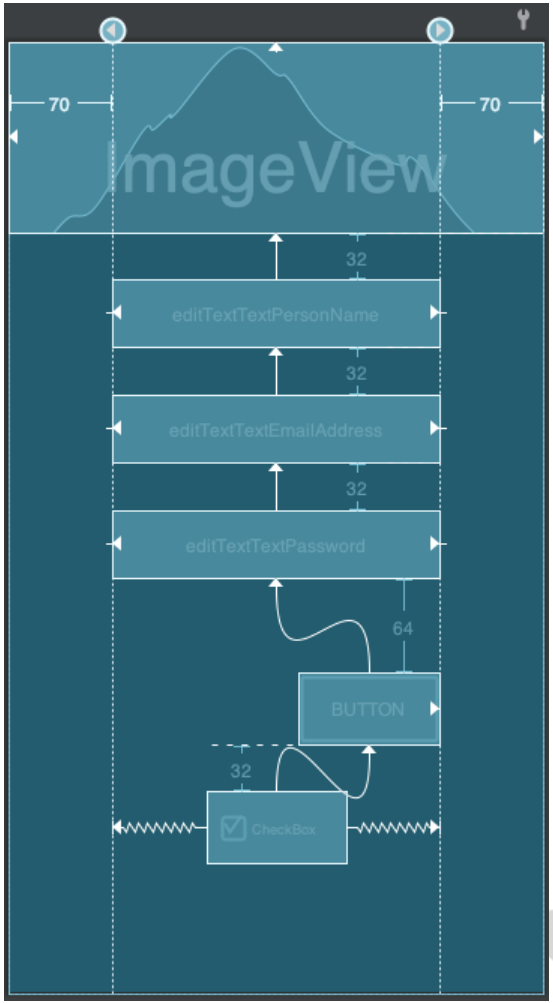
{desafío}
latam_

Ejercicio

Constraint Layout

Construir el siguiente diseño:

- ImageView debe ocupar todo el ancho de la pantalla.
- Recuerda extraer las dimensiones al archivo dimens.xml (32dp, 64dp, 70dp).
- Los EditText están entre las guidelines y usan width=0dp para ocupar todo el espacio disponible.



¿Cuáles son los componentes necesarios para que la app muestre su primera pantalla y que se vea correctamente?

Próxima sesión...

- *Utilizar los principales componentes de interfaz distinguiendo su uso en un proyecto Android Studio.*

{desafío}
latam_

{desafío}
latam_

Academia de
talentos digitales