TEMA 1. Navegación entre pantallas

Práctica 1 ConstraintLayout

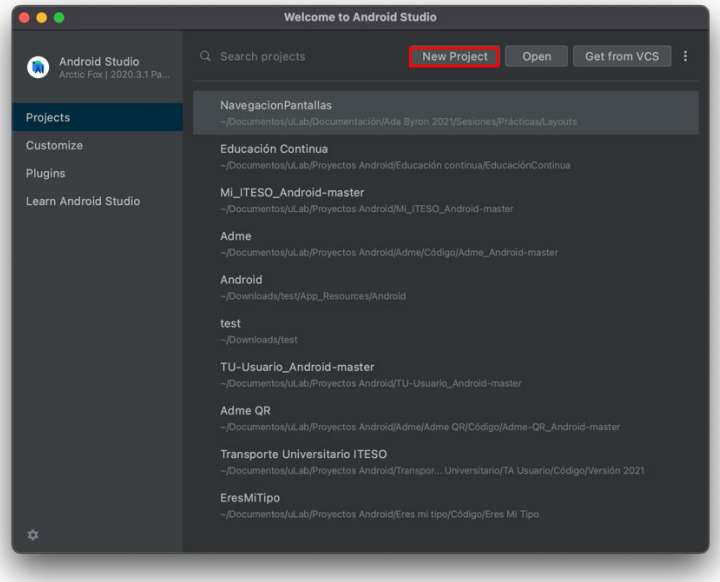
Objetivo específico	Conocer en qué casos podemos usar constraintLayout
Recursos	Android Studio XML
Actividades	Crearemos un activity que contenga un constraintLayout
Producto(s)	Al final de la práctica deberá mostrar una vista en la que mostramos varios objetos posicionados y que se pueda mostrar en modo portrait (vertical) y modo landscape (horizontal)

PASO 1. Abrir Android Studio.

Abrir Android Studio

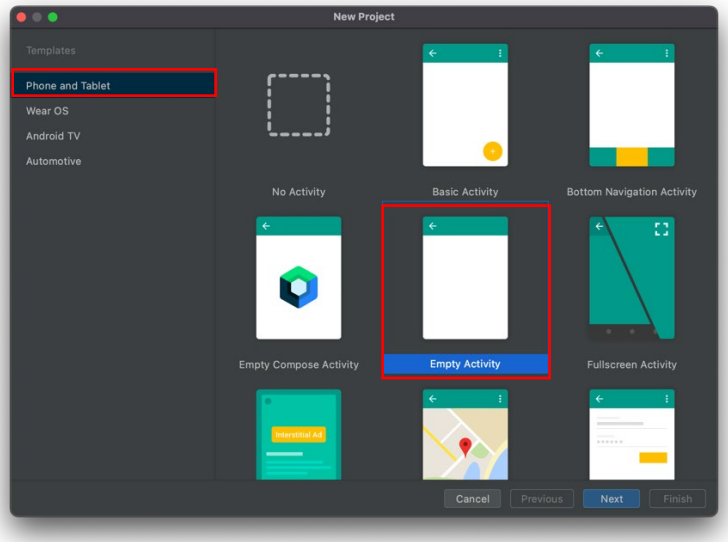
PASO 2. Elegir nuevo proyecto

Seleccionar New project



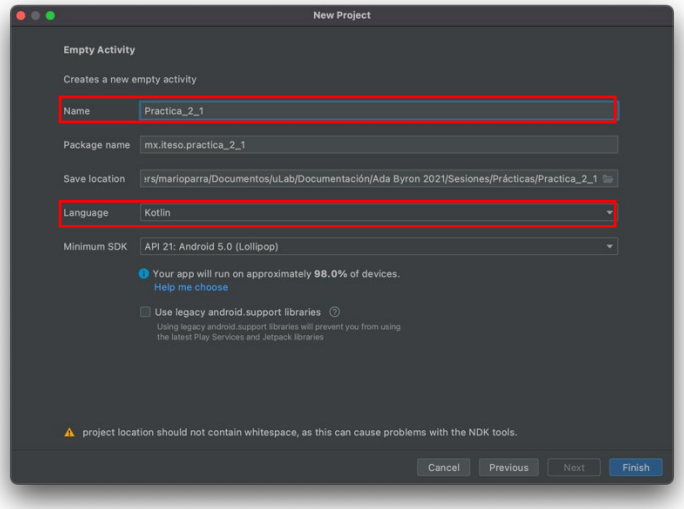
PASO 3. Elegir Empty Activity

Seleccionar Phone and Tablet, luego seleccionar Empty Activity, y dar click en Next



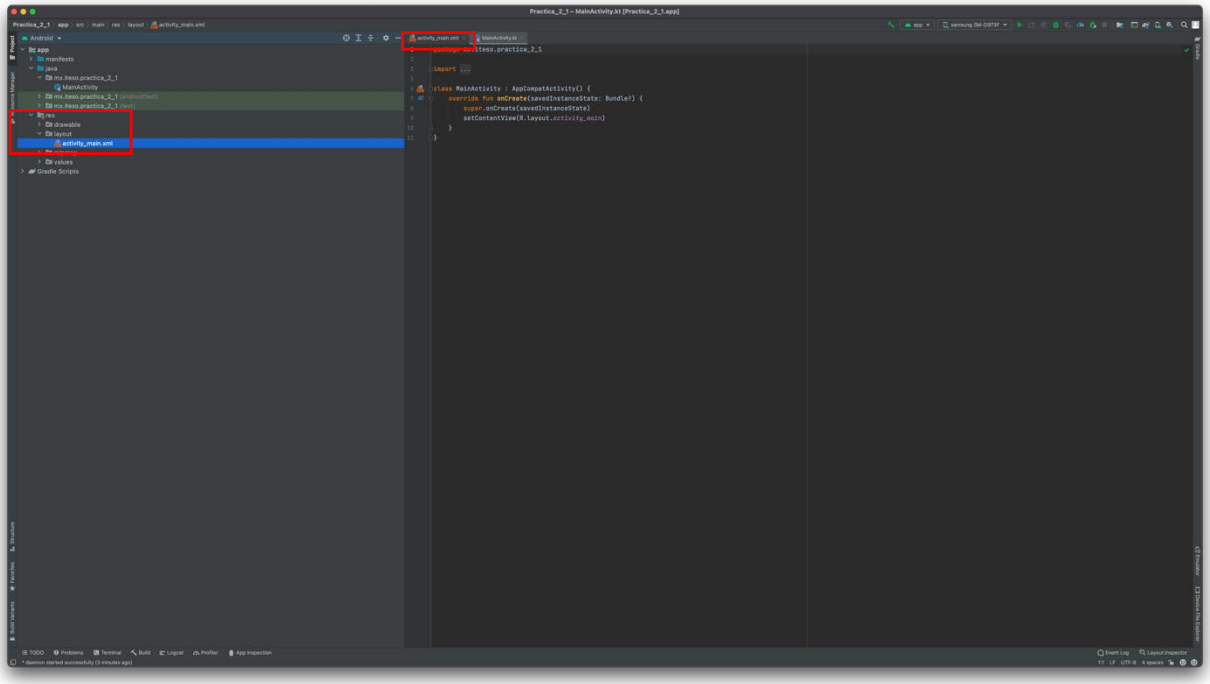
PASO 4. Configuración del nuevo proyecto

En nombre escribir Practica_2_1 y seleccionar el lenguaje Kotlin, después dar click en Finish



PASO 5. Abrir la clase activity_main.xml

Abrir la clase activity_main.xml, en caso de que no esté abierto ir a la estructura de Android, abrir la carpeta res, luego layout y ahí vendrán todas nuestras activities



PASO 6. Seleccionar Split

Seleccionar Split

PASO 7. Crearemos nuestro ConstraintLayout

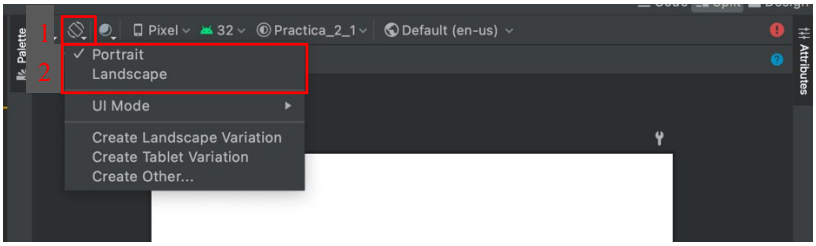
Con el siguiente código agregaremos dos ImageView y un button.

```
<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=".MainActivity">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        tools:layout_editor_absoluteX="64dp"
        tools:layout_editor_absoluteY="123dp"
        tools:srcCompat="@tools:sample/avatars" />

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        tools:layout_editor_absoluteX="216dp"
        tools:layout_editor_absoluteY="123dp"
        tools:srcCompat="@tools:sample/avatars" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        tools:layout_editor_absoluteX="158dp"
        tools:layout_editor_absoluteY="306dp" />

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

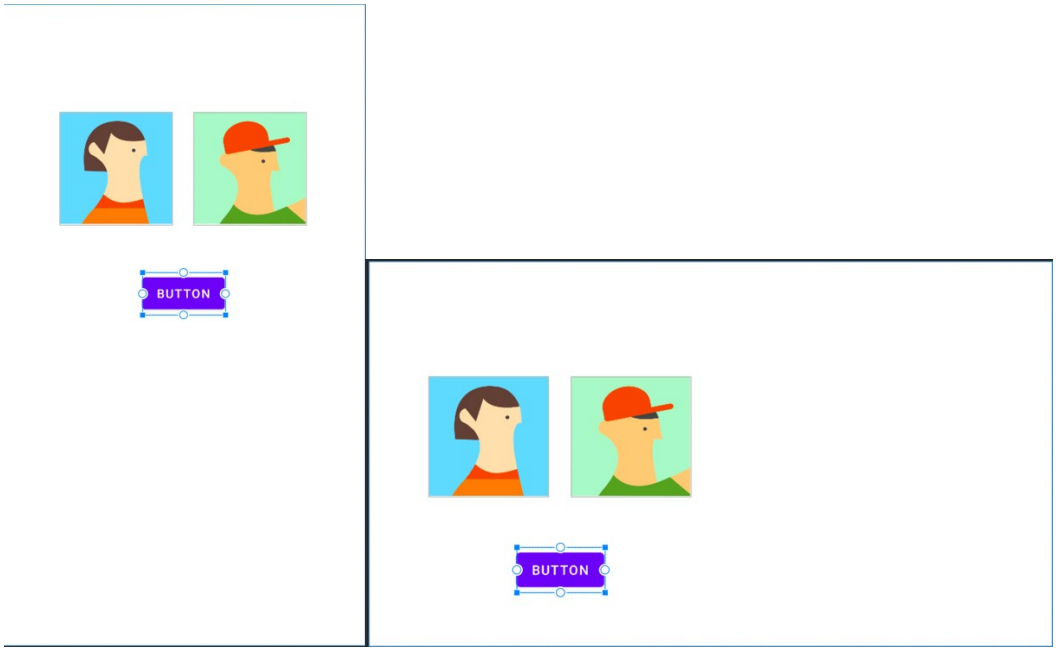
Como podemos ver en el código los tres objetos tienen el atributo ***layout_editor_absoluteX*** y ***layout_editor_absoluteY*** que es la posición donde se posicionará el objeto en la vista. Si arrastramos el objeto por la vista podremos ver que estos valores cambiarán.

PASO 7. Landscape

Para cambiar la posición de la vista debemos dar click aquí.



Resultado.



Práctica 2 FrameLayout

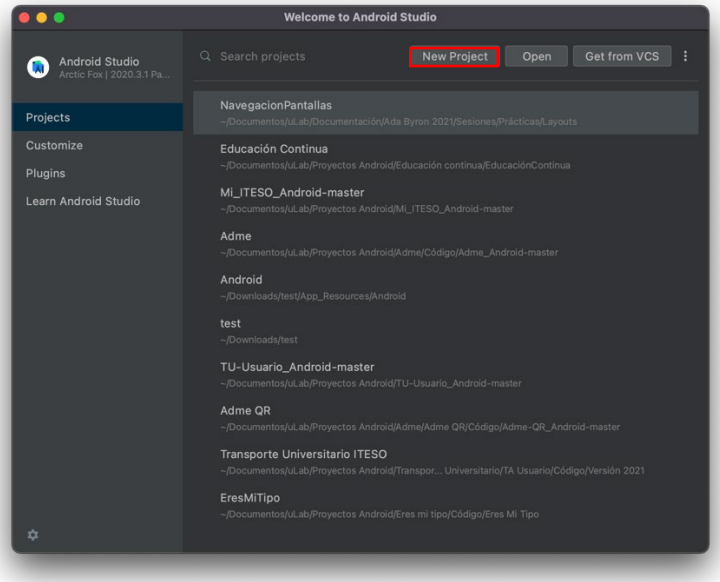
Objetivo específico	Conocer en qué casos podemos usar frameLayout
Recursos	Android Studio XML
Actividades	Crearemos un activity que contenga un frameLayout
Producto(s)	Al final de la práctica deberá mostrar una vista en la que mostramos varios objetos posicionados en un punto específico.

PASO 1. Abrir Android Studio.

Abrir Android Studio

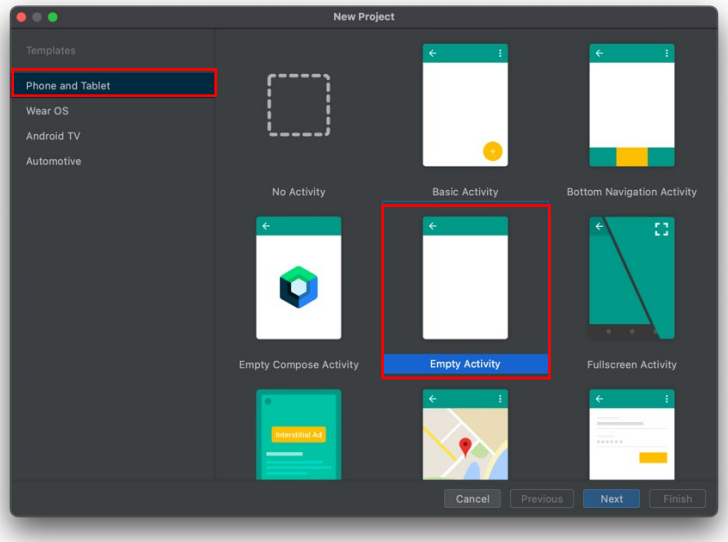
PASO 2. Elegir nuevo proyecto

Seleccionar New project



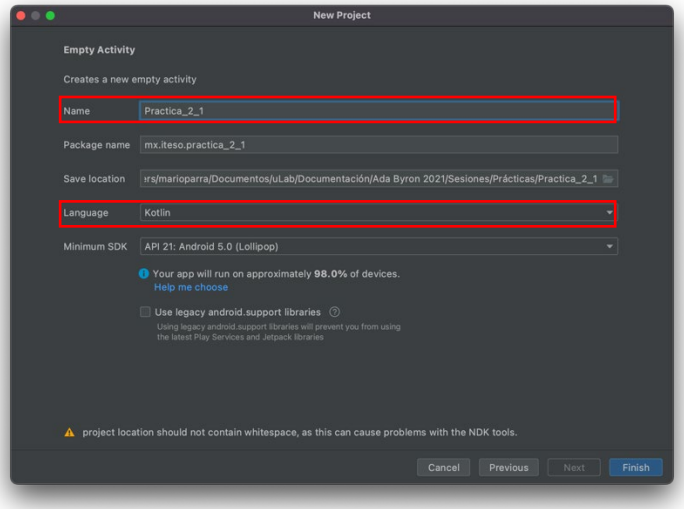
PASO 3. Elegir Empty Activity

Seleccionar Phone and Tablet, luego seleccionar Empty Activity, y dar click en Next



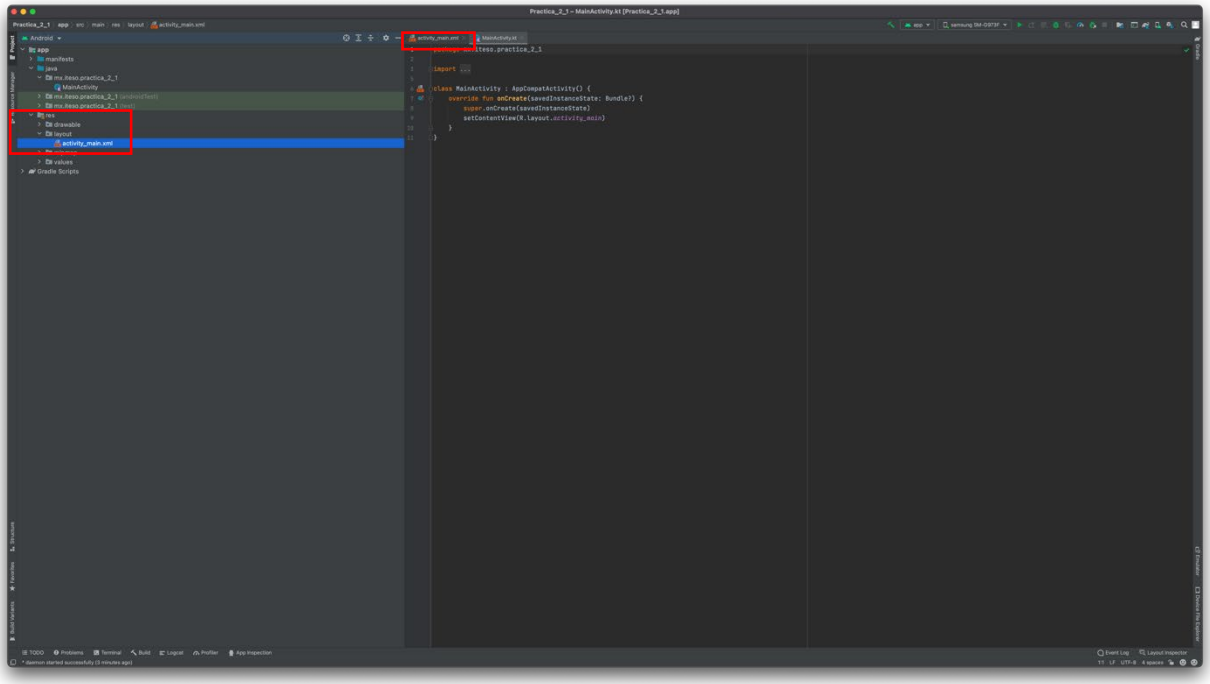
PASO 4. Configuración del nuevo proyecto

En nombre escribir Practica_2_2 y seleccionar el lenguaje Kotlin, después dar click en Finish



PASO 5. Abrir la clase activity_main.xml

Abrir la clase activity_main.xml, en caso de que no esté abierto ir a la estructura de Android, abrir la carpeta res, luego layout y ahí vendrán todas nuestras activities



PASO 6. Seleccionar Split

Seleccionar Split

PASO 7. Crearemos nuestro FrameLayout

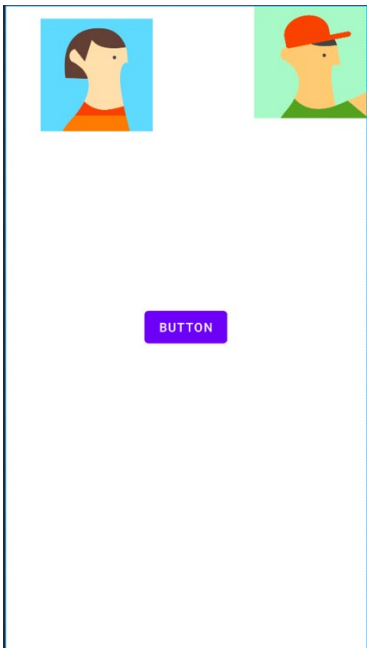
Agregaremos una imagen con margen a la izquierda de 40dp y un margen de arriba de 15dp. La segunda imagen la pondremos en la esquina superior derecha y un botón en el centro de la vista.

```
<FrameLayout
    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=".MainActivity">


    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="40dp"
        android:layout_marginTop="15dp"
        tools:srcCompat="@tools:sample/avatars" />

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="end|end"
        tools:srcCompat="@tools:sample/avatars"/>

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button"
        android:layout_gravity="center"/>

</FrameLayout>
```

Como veremos el atributo que usamos para dar márgenes es ***android:layout_margin*** y solo cambiamos el final para saber a donde le pondremos el margen por ejemplo

android:layout_marginLeft para poner el margen izquierdo seguido del valor númerico y el prefijo dp (densidad de pixeles) ejemplo 40dp significa que el margen izquierdo será de 40.

Resultado.



Práctica 3 LinearLayout

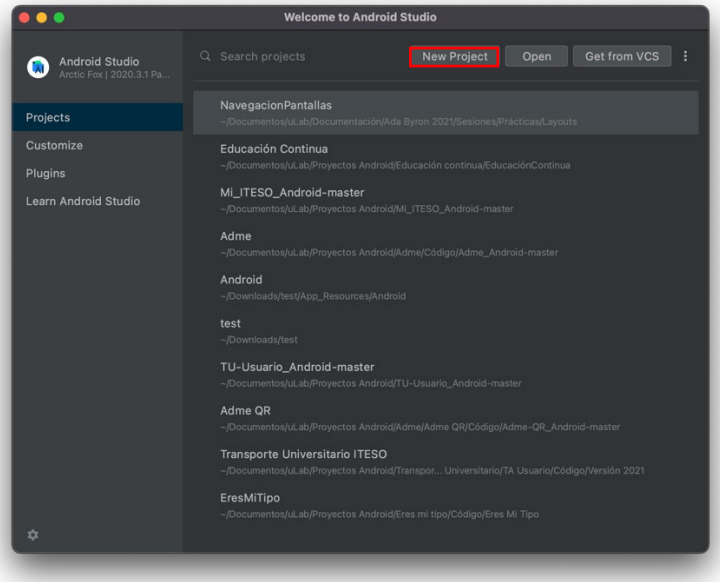
Objetivo específico	Conocer en qué casos podemos usar linearLayout
Recursos	Android Studio XML
Actividades	Crearemos un activity que contenga un linearLayout
Producto(s)	Al final de la práctica deberá mostrar una vista en la que mostramos varios objetos orientados de forma horizontal u vertical.

PASO 1. Abrir Android Studio.

Abrir Android Studio

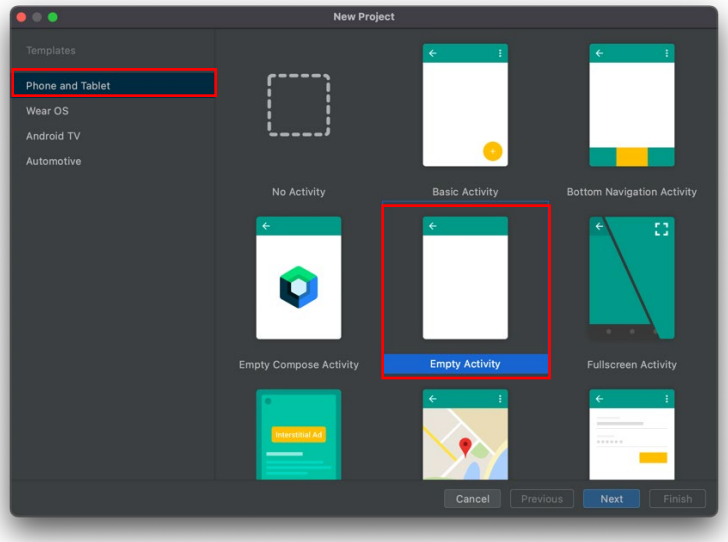
PASO 2. Elegir nuevo proyecto

Seleccionar New project



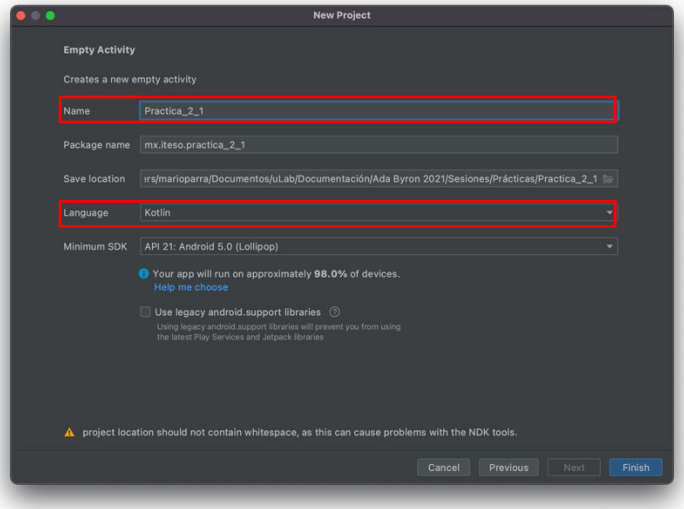
PASO 3. Elegir Empty Activity

Seleccionar Phone and Tablet, luego seleccionar Empty Activity, y dar click en Next



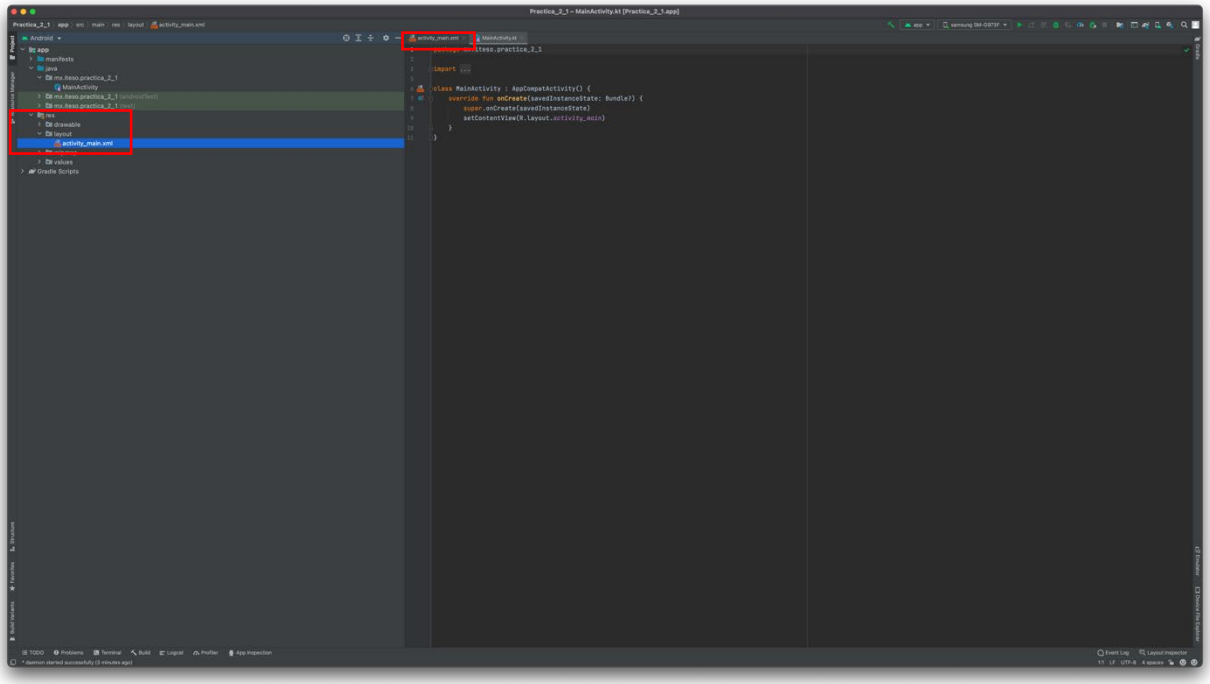
PASO 4. Configuración del nuevo proyecto

En nombre escribir Practica_2_3 y seleccionar el lenguaje Kotlin, después dar click en Finish



PASO 5. Abrir la clase activity_main.xml

Abrir la clase activity_main.xml, en caso de que no esté abierto ir a la estructura de Android, abrir la carpeta res, luego layout y ahí vendrán todas nuestras activities



PASO 6. Seleccionar Split

Seleccionar Split

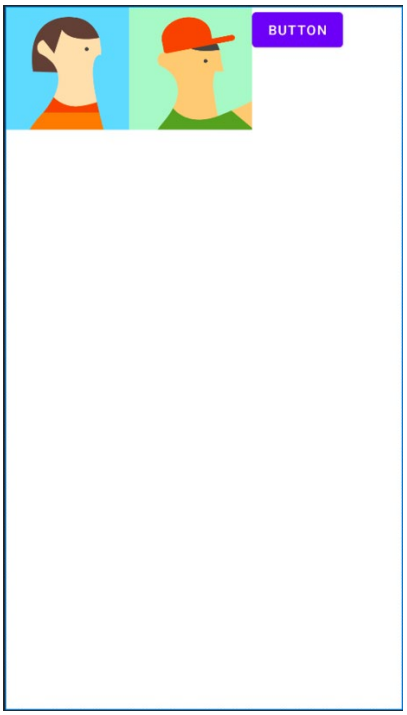
PASO 7. Crearemos nuestro LinearLayout

Agregaremos dos imágenes y un botón con orientación horizontal.

```
<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">


    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        tools:srcCompat="@tools:sample/avatars" />

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        tools:srcCompat="@tools:sample/avatars"/>

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

</LinearLayout>
```

Este layout es más sencillo y con el atributo android:orientation="horizontal" le damos la orientación horizontal y con esto los objetos se mostrarán en la vista juntos y se acomodarán de izquierda a derecha.

Resultado.


BUTTON

Práctica 4 RelativeLayout

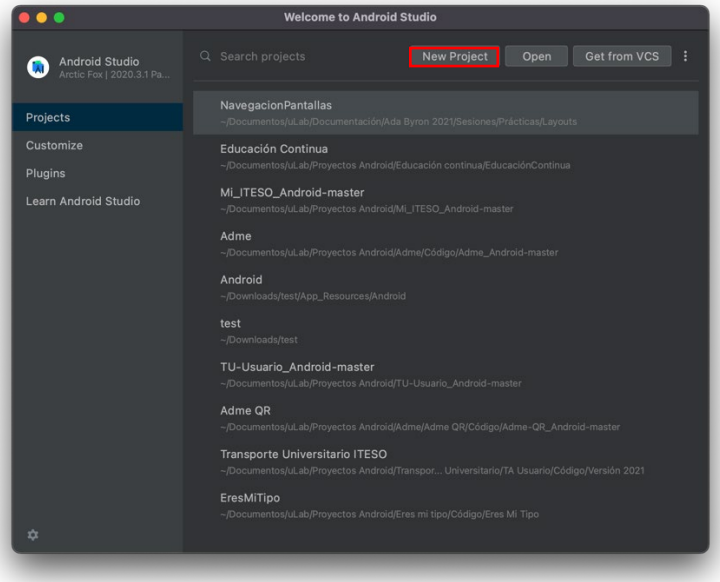
Objetivo específico	Conocer en qué casos podemos usar relativeLayout
Recursos	Android Studio XML
Actividades	Crearemos un activity que contenga un relativeLayout
Producto(s)	Al final de la práctica deberá mostrar una vista en la que mostramos varios objetos donde podemos acomodarlos donde nos guste.

PASO 1. Abrir Android Studio.

Abrir Android Studio

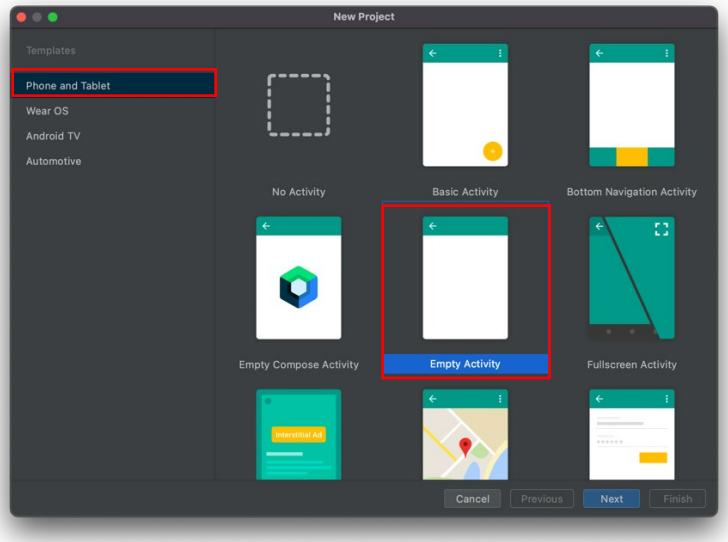
PASO 2. Elegir nuevo proyecto

Seleccionar New project



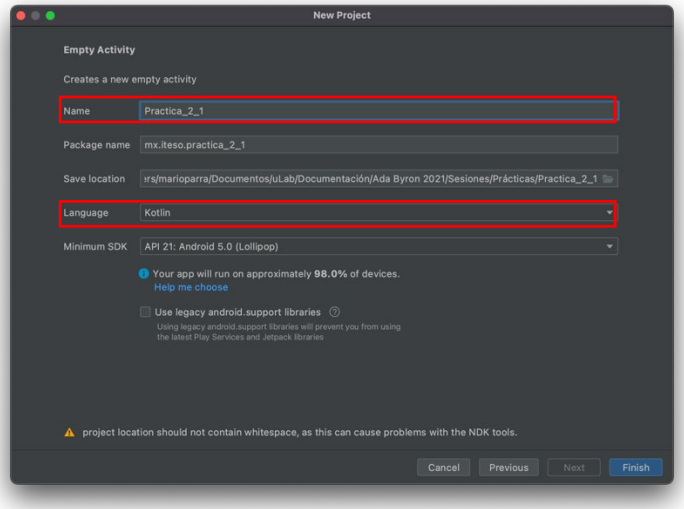
PASO 3. Elegir Empty Activity

Seleccionar Phone and Tablet, luego seleccionar Empty Activity, y dar click en Next



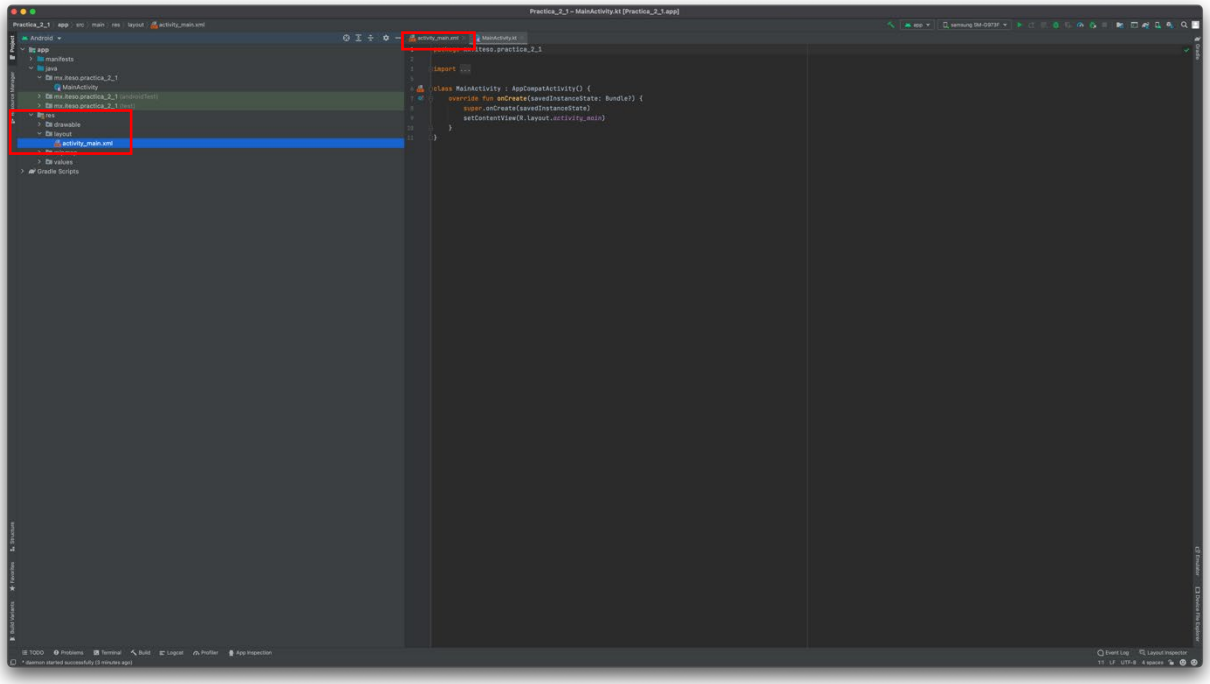
PASO 4. Configuración del nuevo proyecto

En nombre escribir Practica_2_4 y seleccionar el lenguaje Kotlin, después dar click en Finish



PASO 5. Abrir la clase activity_main.xml

Abrir la clase activity_main.xml, en caso de que no esté abierto ir a la estructura de Android, abrir la carpeta res, luego layout y ahí vendrán todas nuestras activities



PASO 6. Seleccionar Split

Seleccionar Split

PASO 7. Crearemos nuestro RelativeLayout

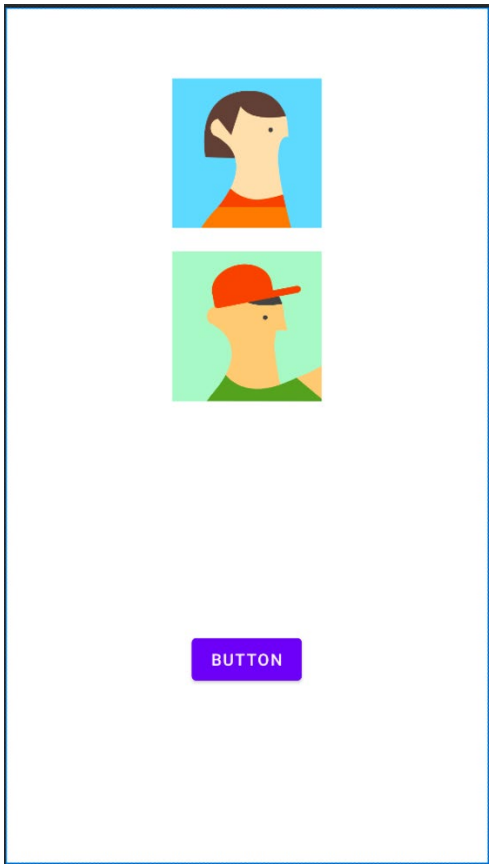
Agregaremos dos imágenes y un botón, la primera imagen tendrá un margen superior anclado al tope de la vista con 60 dp, la segunda imagen estará anclada a la parte inferior de la primera imagen con 20 dp. Para finalizar el botón estará anclado al margen inferior de la vista con 150 dp.

```
<RelativeLayout
    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=".MainActivity">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="60dp"
        tools:srcCompat="@tools:sample/avatars" />

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/imageView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        tools:srcCompat="@tools:sample/avatars" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_marginBottom="150dp"
        android:layout_centerHorizontal="true"
        android:text="Button"
        android:layout_gravity="center"/>
```

```
</RelativeLayout>
```

Como vemos en el código el atributo android:layout_alignParentTop="true" nos dice que estamos anclando la primera imagen al tope de la vista y podemos darle un margen para dar el diseño que se desea con el atributo marginTop. Con la segunda imagen tenemos el atributo android:layout_below="@+id/imageView" donde decimos que queremos anclarnos a la parte inferior de la primera imagen. Aquí vemos algo nuevo @+id/imageView con esto nos referimos que estamos llamando al objeto con el id imageView, para esto necesitamos agregarle un id al objeto que deseamos buscar como lo vemos en los tres objetos que tenemos con el atributo android:id="@+id/nombre_objeto". En el boton vemos android:layout_alignParentBottom="true" que igual que en la primera imagen esta vez anclamos el boton a la parte inferior de la vista con 150dp.

Resultado.



Práctica 5 Navegación entre pantallas

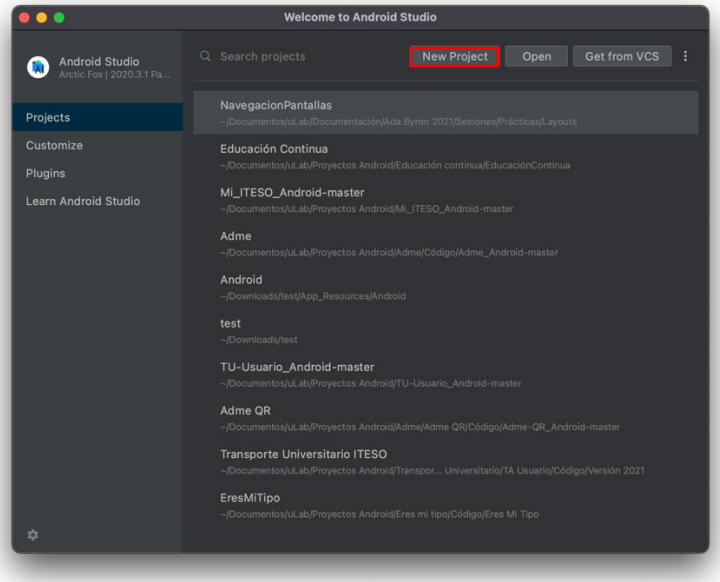
Objetivo específico	Navegar de la vista 1 a la vista 2
Recursos	Android Studio XML Kotlin
Actividades	Crearemos un segundo activity donde reciba información de main activity
Producto(s)	Al final de la práctica deberá enviar información del main activity al segundo activity y mostrar el resultado.

PASO 1. Abrir Android Studio.

Abrir Android Studio

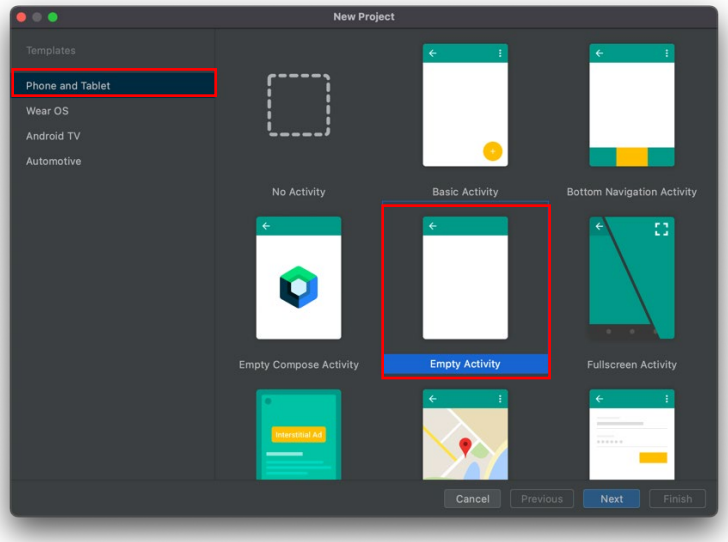
PASO 2. Elegir nuevo proyecto

Seleccionar New project



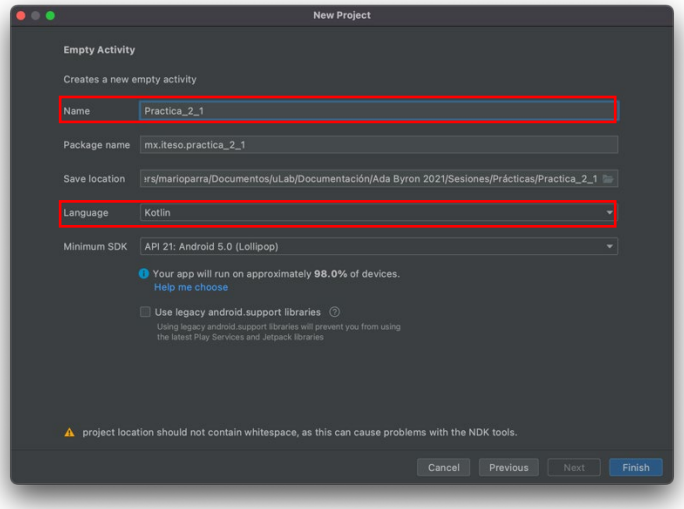
PASO 3. Elegir Empty Activity

Seleccionar Phone and Tablet, luego seleccionar Empty Activity, y dar click en Next



PASO 4. Configuración del nuevo proyecto

En nombre escribir Practica_2_4 y seleccionar el lenguaje Kotlin, después dar click en Finish



PASO 5. Abrir la clase activity_main.xml

Abrir