UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERA MECANICA ELECTRICA, ELECTRONICA Y DE SISTEMAS.

ESCUELA PROFESIONAL DE INGENIERIA DE SISTEMAS

INFORME DE IMPLEMENTACION DE CONSTRAINTLAOUYT

ESTUDIANTE:

APAZA RAMIREZ JIMMY EDSON

ASIGNATURA:

DESARROLLO BASADO DEN PLATAFORMAS II

CICLO: “IV” GRUPO: “C”

DOCENTE:

ING. RUELAS ACERO DONIA ALIZANDRA

PUNO - PERU

2024

Contenido del Informe

I. Informe Técnico sobre la Implementación de ConstraintLayout en Jetpack Compose

Introducción

En el ámbito del desarrollo de aplicaciones móviles contemporáneas, la interfaz de usuario (UI) es esencial para asegurar una experiencia de usuario satisfactoria. Jetpack Compose, la herramienta moderna de UI para Android, proporciona una forma declarativa y eficiente de construir interfaces. Dentro de Compose, ConstraintLayout facilita el diseño de interfaces complejas estableciendo relaciones de restricción entre los componentes. Este informe examina tres diferentes implementaciones de ConstraintLayout en Jetpack Compose, resaltando sus objetivos, metodologías y funcionalidades particulares.

Objetivo

El propósito de este informe es analizar y detallar tres fragmentos de código que utilizan ConstraintLayout en Jetpack Compose. Se busca entender cómo cada fragmento estructura la interfaz de usuario, identificar las diferencias y similitudes entre ellos, y explorar las técnicas empleadas para manejar las restricciones y la disposición de los elementos gráficos.

Metodología

Para realizar este análisis, se revisaron tres implementaciones distintas de ConstraintLayout en Jetpack Compose. Cada fragmento de código fue examinado de forma individual para identificar sus componentes, la estructura de las restricciones y el comportamiento general. Posteriormente, se compararon las similitudes y diferencias entre las implementaciones, ofreciendo una explicación clara de cómo cada una contribuye a la disposición de los elementos gráficos en la UI.

Análisis de Implementaciones

Primera Implementación: ConstraintExample con Cinco Cajas

Descripción General:

La primera implementación define una función composable llamada ConstraintExample que emplea ConstraintLayout para organizar cinco cajas (Box) de diferentes colores en la pantalla. Cada caja posee un tamaño fijo y una posición relativa basada en restricciones específicas.

Componentes y Funcionamiento:

ConstraintLayout Principal:

Se extiende para ocupar todo el espacio disponible (Modifier.fillMaxSize()).

Se crean cinco referencias (box1 a box5) para posicionar las cajas.

Cajas:

Box1 (Rojo):

Tamaño: 120 dp.

Color: Rojo.

Restricción: Centrado respecto al contenedor padre (centerTo(parent)).

Box2 (Azul):

Tamaño: 120 dp.

Color: Azul.

Restricciones:

Vinculada al fondo de Box1 (bottom.linkTo(box1.top)).

Vinculada al inicio de Box1 (end.linkTo(box1.start)).

Box3 (Magenta):

Tamaño: 120 dp.

Color: Magenta.

Restricciones:

Vinculada al fondo de Box1.

Vinculada al final de Box1 (start.linkTo(box1.end)).

Box4 (Amarillo):

Tamaño: 120 dp.

Color: Amarillo.

Restricciones:

Vinculada a la parte superior de Box1 (top.linkTo(box1.bottom)).

Vinculada al inicio de Box1.

Box5 (Verde):

Tamaño: 120 dp.

Color: Verde.

Restricciones:

Vinculada a la parte superior de Box1.

Vinculada al final de Box1.

Conclusiones de la Primera Implementación:

Este diseño establece una disposición simétrica alrededor de la caja central roja (Box1). Las cajas azul y magenta se sitúan arriba a la izquierda y a la derecha, respectivamente, mientras que las cajas amarilla y verde se ubican debajo de la caja central, también a los lados. Este patrón crea una estructura equilibrada y comprensible, adecuada para interfaces que requieren una disposición simétrica de sus elementos.

Segunda Implementación: ConstraintExample con Siete Cajas

Descripción General:

La segunda implementación amplía la anterior añadiendo dos cajas adicionales (Box6 y Box7). Esto permite una disposición más compleja y detallada de los elementos gráficos en la interfaz.

Componentes y Funcionamiento:

ConstraintLayout Principal:

Similar a la primera implementación, ocupa todo el espacio disponible.

Se crean siete referencias (box1 a box7).

Cajas:

Boxes 1 a 5:

Funcionan de manera similar a la primera implementación, con ligeras variaciones en los colores (por ejemplo, Box5 es Cyan en lugar de Verde).

Box6 (Verde):

Tamaño: 120 dp.

Color: Verde.

Restricciones:

Vinculada al fondo de Box2.

Centrada horizontalmente respecto al contenedor padre.

Box7 (Negro):

Tamaño: 120 dp.

Color: Negro.

Restricciones:

Vinculada a la parte superior de Box4.

Centrada horizontalmente respecto al contenedor padre.

Conclusiones de la Segunda Implementación:

La incorporación de Box6 y Box7 introduce más puntos de referencia en la disposición, permitiendo una estructura más dinámica y compleja. Box6 y Box7 están alineadas horizontalmente con respecto al contenedor padre, pero están vinculadas a Box2 y Box4, respectivamente. Esto genera una jerarquía visual más rica, útil para interfaces que requieren múltiples niveles de disposición o elementos adicionales alrededor de una estructura central.

Tercera Implementación: ConstraintExample con Caja Contenedora

Descripción General:

La tercera implementación introduce una containerBox que a su vez contiene un ConstraintLayout anidado con tres cajas más pequeñas. Este enfoque permite la creación de subestructuras dentro de la disposición principal.

Componentes y Funcionamiento:

ConstraintLayout Principal:

Similar a las implementaciones anteriores, ocupa todo el espacio disponible.

Se crean seis referencias (box1 a box5 y containerBox).

Cajas Principales (1 a 5):

Funcionan de manera similar a la primera implementación, con algunas variaciones en los colores.

ContainerBox:

Tamaño: 120 dp.

Sin color de fondo específico.

Restricciones:

Vinculada al fondo de Box1.

Centrada horizontalmente respecto a Box1.

ConstraintLayout Anidado Dentro de ContainerBox:

Ocupa todo el espacio de containerBox.

Se crean tres referencias (miniBox1, miniBox2, miniBox3).

MiniBoxes:

MiniBox1 (Negro):

Tamaño: 40 dp.

Color: Negro.

Restricciones:

Vinculada a la parte superior y al final del contenedor.

MiniBox2 (Negro):

Tamaño: 40 dp.

Color: Negro.

Restricciones:

Centrada dentro del contenedor.

MiniBox3 (Cyan):

Tamaño: 40 dp.

Color: Cyan.

Restricciones:

Vinculada a la parte inferior y al inicio del contenedor.

Conclusiones de la Tercera Implementación:

Este diseño añade una capa adicional de complejidad al incorporar un ConstraintLayout anidado dentro de containerBox. Las tres mini cajas dentro del contenedor permiten una

disposición más detallada dentro de una área específica de la interfaz. Esta técnica es útil para agrupar elementos relacionados y gestionar subdisposiciones sin complicar la estructura principal.

Conclusiones Generales

Las tres implementaciones analizadas demuestran la flexibilidad y el poder de ConstraintLayout en Jetpack Compose para diseñar interfaces de usuario complejas y dinámicas. Mientras que la primera implementación establece una base simétrica y sencilla, la segunda añade complejidad con más elementos y relaciones de restricción. La tercera implementación, por su parte, muestra cómo anidar ConstraintLayout para crear subdisposiciones dentro de una estructura mayor.

Estas implementaciones destacan la importancia de planificar cuidadosamente las relaciones de restricción para lograr interfaces equilibradas y funcionales. Además, evidencian cómo la reutilización y la modularidad pueden mejorar la mantenibilidad y escalabilidad del código de la interfaz de usuario.

Recomendaciones

Modularidad: Considerar el uso de composables anidados para gestionar disposiciones complejas, tal como se ejemplifica en la tercera implementación.

Consistencia de Colores y Tamaños: Mantener una paleta de colores y tamaños coherente para asegurar una apariencia unificada y profesional.

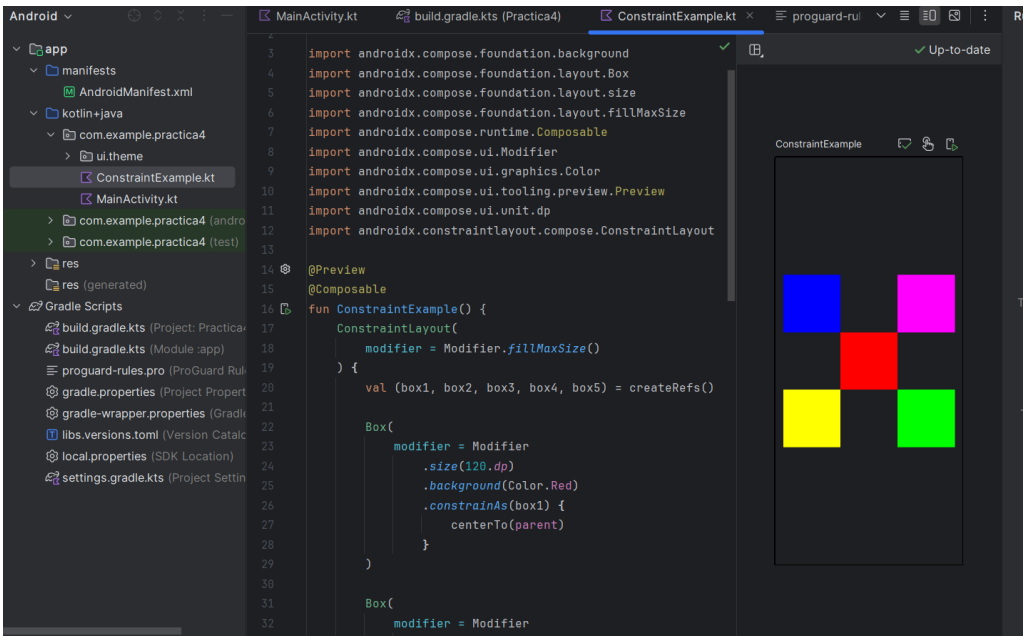
Optimización de Restricciones: Evitar restricciones innecesarias que puedan complicar la disposición y afectar el rendimiento.

Anexos

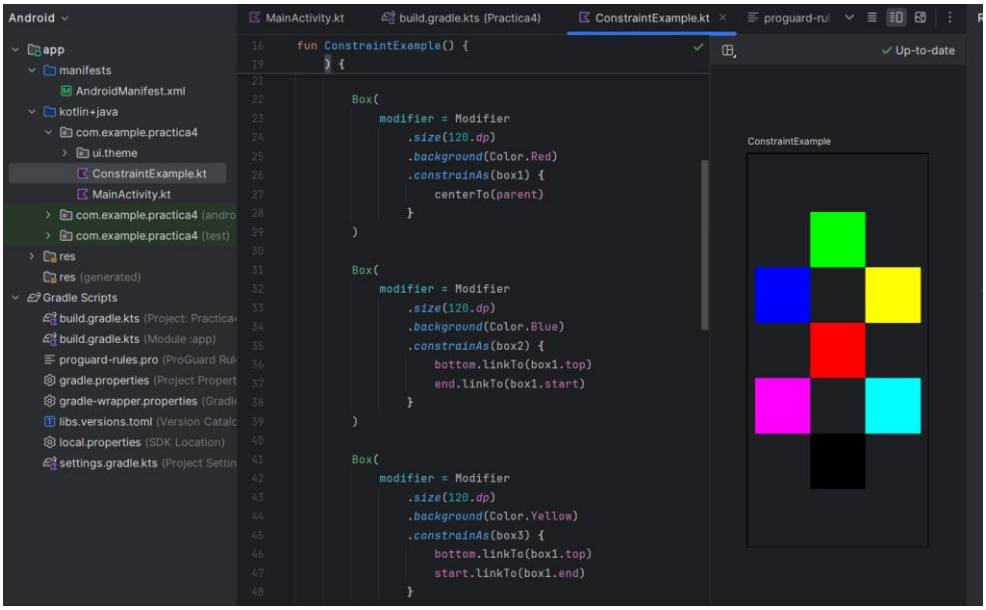
A continuación, se incluyen pruebas que demuestran el funcionamiento de las implementaciones analizadas:

Capturas de Pantalla:

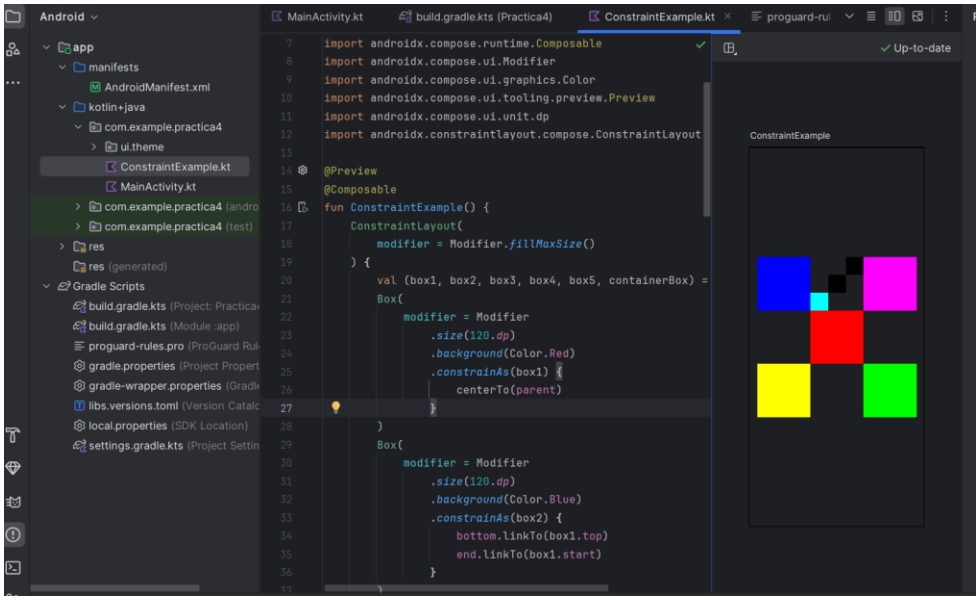
Primera Implementación: Imagen mostrando las cinco cajas organizadas simétricamente.



Segunda Implementación: Imagen con las siete cajas, destacando la adición de Box6 y Box7.



Tercera Implementación: Imagen que muestra la caja contenedora con las mini cajas anidadas.



Los códigos se encuentran adjuntos en el siguiente link:

https://github.com/Jimmnoh/INFORMES-DBPII.git

Estos anexos proporcionan evidencia tangible del funcionamiento y las características de cada implementación, facilitando una mejor comprensión y validación de los análisis presentados.