Taller Android Studio.

Presentado por:

Yordy Erik Nuñez Pineda.

Presentado a:

Carlo Julio Cadena Sarasty.

Centro de la Industria, la Empresa y los Servicios

Ficha:2694667 – Fecha:28/05/2024

Neiva – Huila

1. LinearLayout (Vertical y Horizontal)

Características Principales:

* Línea los elementos de forma horizontal o vertical.
* Distribuye los elementos uno tras otro.

Organización de Objetos:

* Vertical: Los elementos se organizan en una columna.
* Horizontal: Los elementos se organizan en una fila.

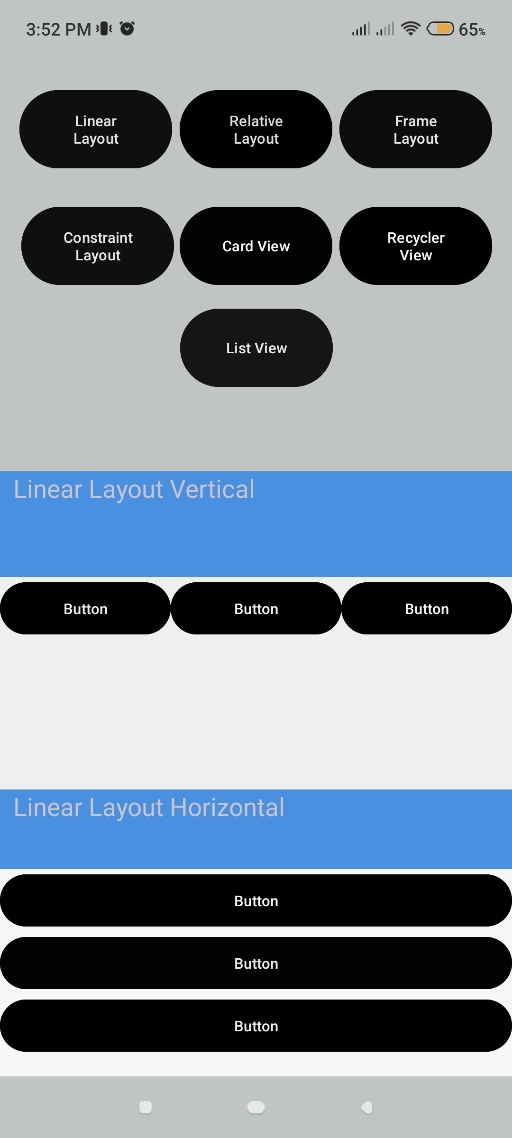
Ventajas:

* Sencillo de usar y entender.
* Facilita la alineación básica de elementos.

Desventajas:

* Puede ser ineficiente con muchos elementos debido a la anidación excesiva.
* Menos flexible para interfaces complejas.

Ejemplo:



1. RelativeLayout

Características Principales:

* Permite posicionar los elementos relativos a otros elementos o al contenedor principal.

Organización de Objetos:

* Los objetos se posicionan en base a relaciones (por ejemplo, "a la derecha de" o "debajo de").

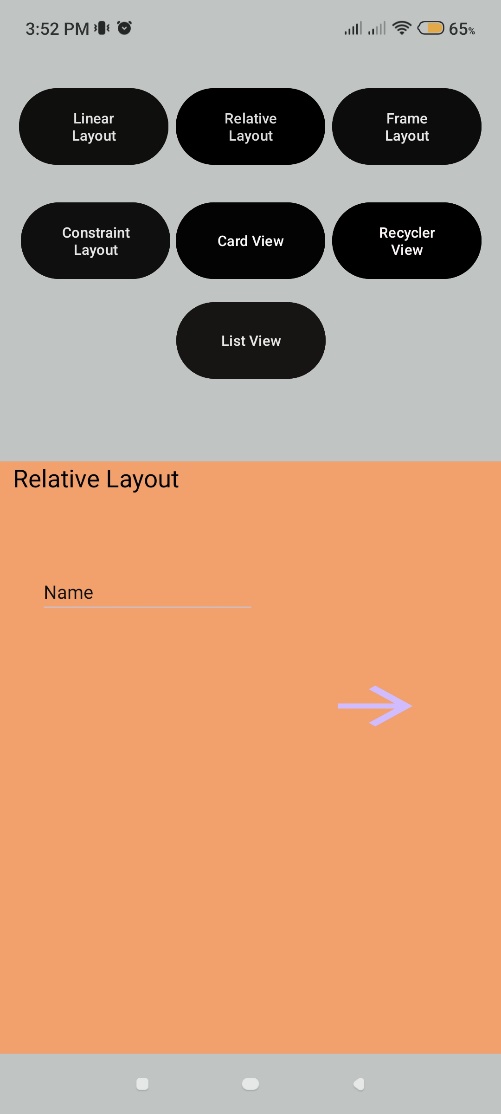
Ventajas:

* Muy flexible para crear interfaces complejas.
* Reduce la necesidad de contenedores anidados.

Desventajas:

* Puede volverse complejo y difícil de mantener con muchas reglas de posicionamiento.
* Puede impactar el rendimiento si se utiliza en exceso.

Ejemplo:



1. FrameLayout

Características Principales:

* Diseñado para contener un solo hijo, aunque puede tener múltiples hijos apilados.

Organización de Objetos:

* Los elementos se apilan uno sobre otro, con el último elemento añadido en la parte superior.

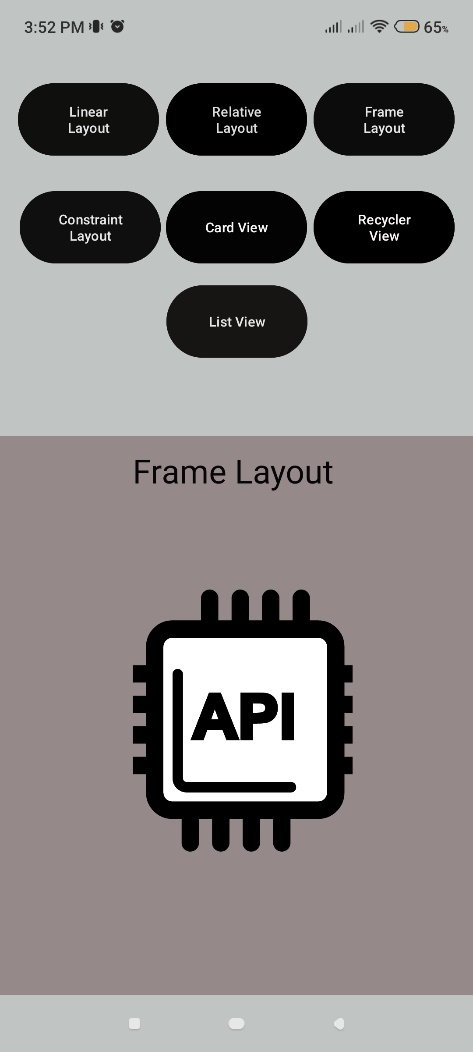
Ventajas:

* Útil para superponer vistas, como imágenes y textos.
* Simple y ligero.

Desventajas:

* No es adecuado para diseños complejos.
* Mantenimiento difícil cuando se manejan múltiples elementos superpuestos.

Ejemplo:



1. ConstraintLayout

Características Principales:

* Permite crear interfaces complejas sin necesidad de anidar múltiples vistas.
* Utiliza un sistema de restricciones para posicionar y dimensionar vistas.

Organización de Objetos:

* Los objetos se posicionan y dimensionan en base a restricciones definidas entre vistas o con el contenedor principal.

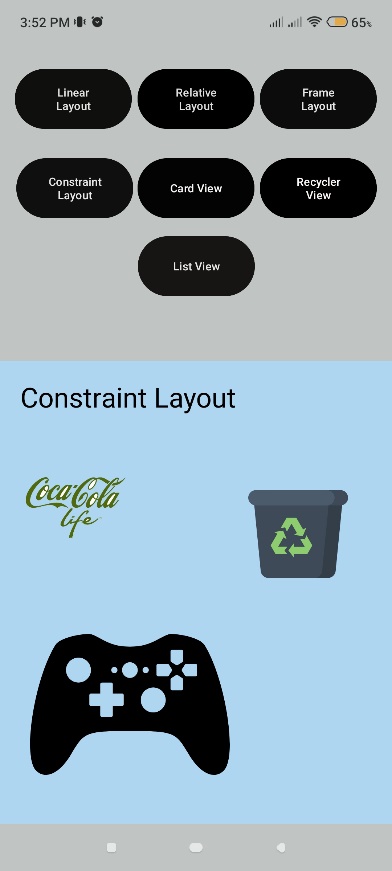
Ventajas:

* Muy flexible y poderoso.
* Mejora el rendimiento al reducir la necesidad de contenedores anidados.

Desventajas:

* Requiere un mayor tiempo de aprendizaje.
* Puede ser más difícil de entender y mantener en comparación con otros layouts simples.

Ejemplo:



1. CardView

Características Principales:

* Contenedor que proporciona un fondo de sombra y bordes redondeados.

Organización de Objetos:

* Puede contener cualquier otro tipo de vista, usualmente un FrameLayout o LinearLayout.

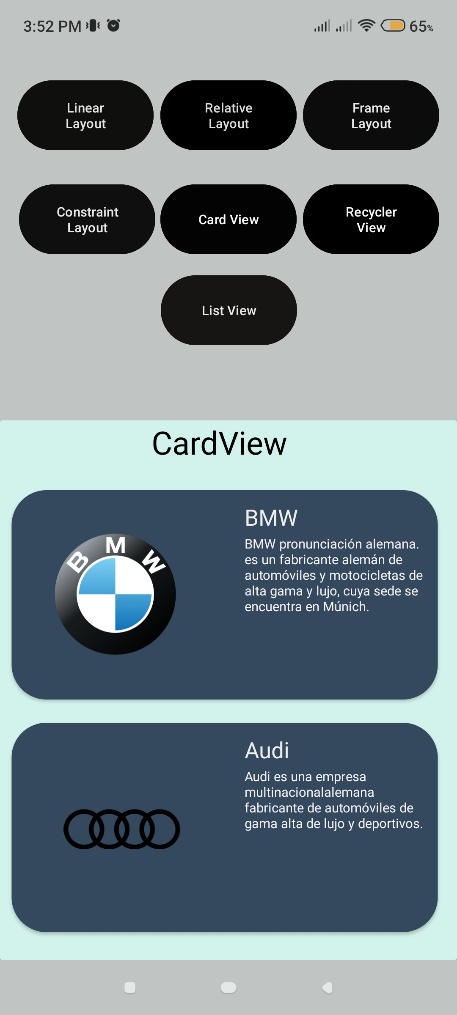
Ventajas:

* Mejora la estética de las aplicaciones al proporcionar un diseño de tarjetas.
* Fácil de usar con RecyclerView para listas atractivas.

Desventajas:

* Puede añadir una sobrecarga de rendimiento debido a las sombras y bordes redondeados.
* No es adecuado para todos los tipos de diseño.

Ejemplo:



1. RecyclerView

Características Principales:

* Diseñado para mostrar grandes cantidades de datos en listas o rejillas de forma eficiente.
* Más flexible y poderoso que ListView.

Organización de Objetos:

* Utiliza un adaptador para enlazar los datos con las vistas.
* Soporta múltiples tipos de layouts (lineal, de rejilla, etc.).

Ventajas:

* Muy eficiente en términos de rendimiento y uso de memoria.
* Soporta animaciones y personalización avanzada.

Desventajas:

* Mayor complejidad en comparación con ListView.
* Requiere más código y configuración inicial.

Ejemplo:

1. ListView

Características Principales:

* Utilizado para mostrar una lista de elementos desplazables.

Organización de Objetos:

* Utiliza un adaptador para enlazar los datos con las vistas.
* Cada elemento de la lista se infla desde una plantilla de diseño.

Ventajas:

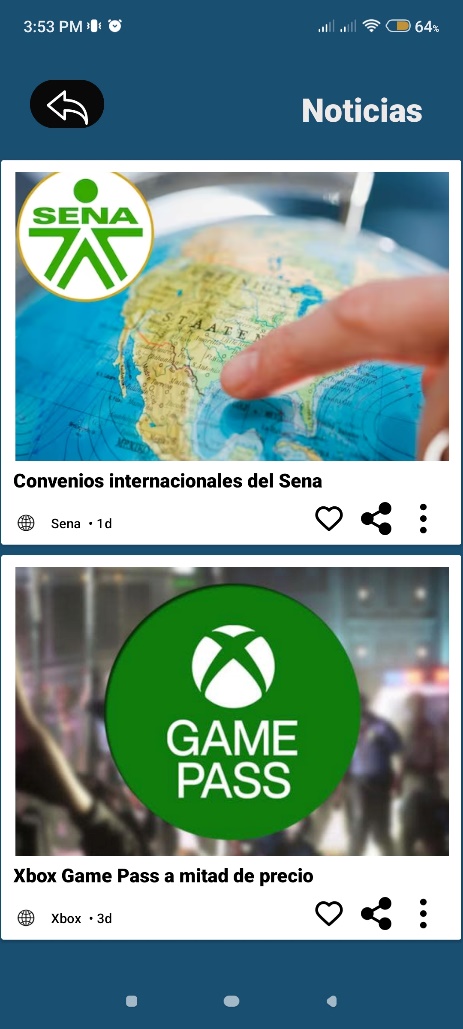
* Simple de usar para listas básicas.
* Soporta selección de elementos y eventos de clic.

Desventajas:

* Menos eficiente que RecyclerView para listas grandes.
* Menos flexible y con menos opciones de personalización.

Ejemplo:

2. Interfaces.



Links de Git:

Contenedores: <https://github.com/TheRoydy/contenedores.git>

Mockups: <https://github.com/TheRoydy/mockups.git>