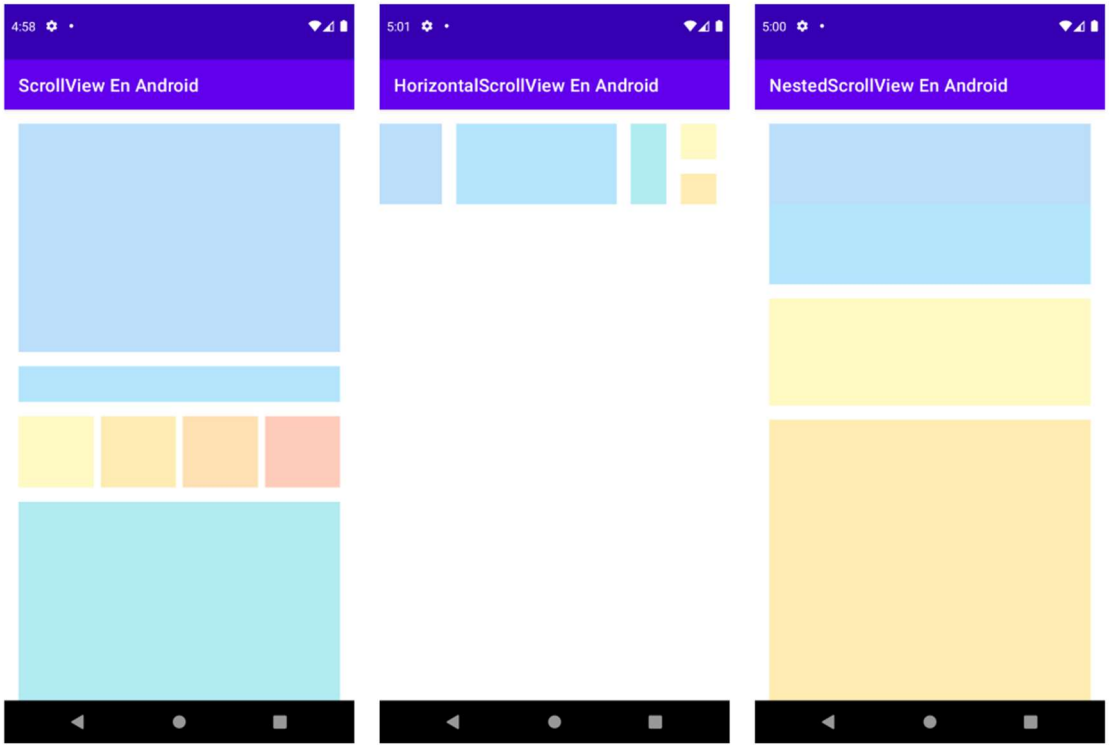
ScrollView

El ScrollView en Android te permite albergar una jerarquía de views con el fin de desplazar su contenido a lo largo de la pantalla, cuando sus dimensiones exceden el tamaño de la misma.

El usuario hará scroll a través de un gesto de swipe vertical u horizontal para revelar la información limitada por el tamaño del contenedor.

Existen los siguientes contenedores:

- **ScrollView** (scroll vertical)
- **HorizonalScrollView** (scroll horizontal)
- **NestedScrollView** (scroll anidados)



Scrolling Vertical

Si la información que deseas proyectar verticalmente es más grande que la orientación de tu pantalla móvil, entonces envuelve el ViewGroup (layout) del contenido con el contenedor **ScrollView**:

ScrollView soporta un solo hijo como contexto de desplazamiento, por lo que la estructura general de tu diseño debe encontrarse en él (pon como hijo un ConstraintLayout, dentro del cual distribuye los controles que quieres que existan en tu página), o bien recubre tu *ViewGroup* principal desde el layout con la etiqueta <ScrollView>.

Es necesario que uses wrap_content en android:layout_height del contenedor directo (ViewGroup o Layout) para ajustar la zona al scrolling.

Puedes añadir este elemento desde Android Studio yendo a Palette > Containers > ScrollView:

Por ejemplo, para probarlo, dentro del layout, añade diferentes textView con una altura fija de 200dp con el fin de desbordar la capacidad de la pantalla.

Scrolling Horizontal

El mismo criterio aplica para los contenidos que abruman horizontalmente la pantalla del dispositivo. Usa la clase **HorizontalScrollView** para proveer desplazamiento entre los límites del contenedor:

Al igual que ScrollView, HorizontalScrollView hereda de la clase FrameLayout, por lo que la jerarquía a scrollear debe ser parte de un solo nodo (pon como hijo un ConstraintLayout, dentro del cual distribuye los controles que quieres que existan en tu página)

El ViewGroup directo del HorizontalScrollView debe asignar wrap_content a su atributo android:layout_width para mantener el dinamismo del deslizamiento interno.

Anidar Scrolls Con NestedScrollView

En el caso que requieras añadir un ScrollView dentro de otro, utiliza directamente **NestedScrollView**.

Este componente puede actuar como padre o hijo en una jerarquía de scrolling. Por ejemplo, en el siguiente caso, el hijo directo del scroll padre es un ConstraintLayout. Y el del child_scroll es un LineaLayout que tiene cuatro etiquetas verticales.

Ocultar Barras De Scroll

Al desplazar el contenido de un widget de scroll se presentan barras de color gris para indicar el estado y avance del scroll.

Las barras de scroll están activadas por defecto, para ocultarlas desde Kotlin usa las propiedades, isHorizontalScrollBarEnabled y isVerticalScrollBarEnabled a false para deshabilitarlas, o bien el atributo XML android:scrollbars con none.