
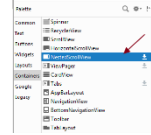
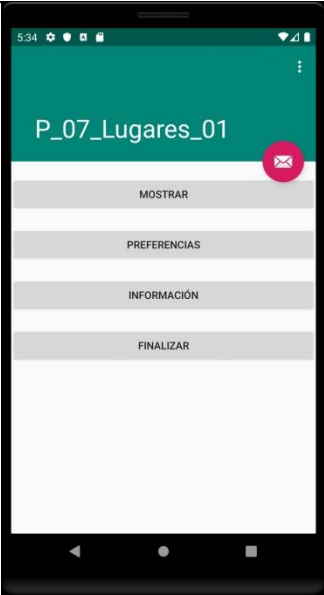



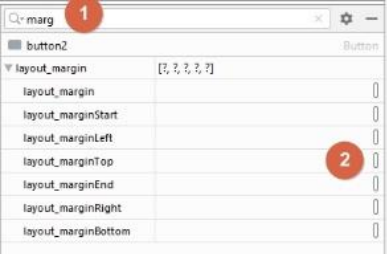
Ejemplo 1: Uso de Temas de la plataforma

1. Crea un nuevo proyecto con nombre P_07_Lugares_01 con plantilla Scrolling Activity.
2. Observa que la plantilla sigue estando basada en Material Design pero usando ahora un tipo de layout propio de Material Design, el NestedScrollView para que el contenido principal pueda desplazarse (por eso, aparece un mensaje  Layout fidelity warning que nos advierte que lo mostrado puede no parecerse a la realidad).
3. **Ejecuta la aplicación** para comprobarlo y observa la animación al desplazar el texto (el botón flotante desaparece y la toolbar se hace más pequeña!).
4. Ya que estamos con esta plantilla que tiene un texto muy largo, abre el fichero strings.xml y observa que en el contenido de la string que contiene dicho texto, está permitido usar \n para indicar un salto de línea.
5. Reemplaza el TextView por un LinearLayout que contenga cuatro Button (un NestedScrollView solo puede contener dentro un elemento, por lo que no puedes introducir directamente los cuatro botones; usando un layout que los contenga se resuelve el problema ya que dentro de un layout puede haber otros!) para conseguir un diseño semejante a la imagen inferior.



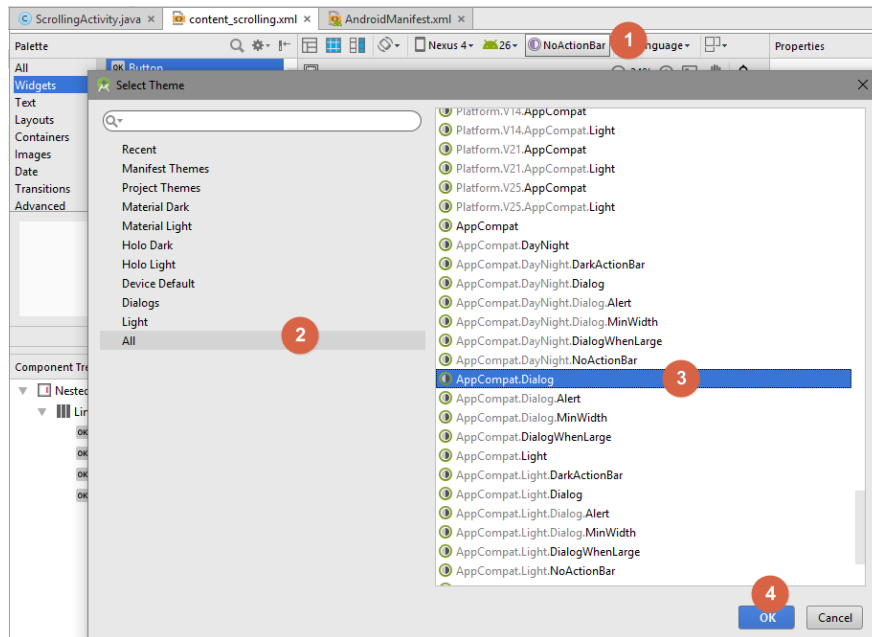


Debes crear recursos en el proyecto:

- Para los textos de los botones (fichero strings.xml). Usa asistente: 
- Para que la dimensión de separación de los botones sea 20dp (fichero dimens.xml). Usa asistente: 

6. Ejecuta la aplicación y cambia la orientación del dispositivo. Podemos ver los botones que no "caben" al usar el desplazamiento, pero es un mal diseño porque no es intuitivo para el usuario dicho desplazamiento.
7. Aplica a la actividad (desde el fichero AndroidManifest.xml), el tema @style/Theme.AppCompat.Dialog. Vuelve a ejecutar. Este tema es utilizado en cuadros de diálogo. No parece muy adecuado para nuestra actividad.

8. Observa que el Editor de Layout, la previsualización no se ha cambiado sola, debemos hacerlo (además no todos funcionan, debe heredar de AppCompatActivity!)



9. Deshaz el cambio realizado en el Manifest y cierra el proyecto (el propósito era que conocieras otro tipo de plantilla).

Ejercicio 2: Uso de Estilos propios

1. En el proyecto P_02_Creciente_01, crea un nuevo estilo con nombre MiEstilo que defina la propiedad color del texto del TextView como naranja, haciendo las modificaciones oportunas en los ficheros:

res/values/colors.xml	res/values/styles.xml
<pre><?xml version="1.0" encoding="utf-8"?> <resources> <color name="colorPrimary">#008577</color> <color name="colorPrimaryDark">#00574B</color> <color name="colorAccent">#D81B60</color> <color name="miNaranja">#FF7F27</color> </resources></pre>	<pre><resources> <!-- Base application theme. --> <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar"> <!-- Customise your theme here. --> <item name="colorPrimary">@color/colorPrimary</item> <item name="colorPrimaryDark">@color/colorPrimaryDark</item> <item name="colorAccent">@color/colorAccent</item> </style> <style name="MiEstilo"> <item name="android:textColor">@color/miNaranja</item> </style> </resources></pre>

2. Aplícalo al TextView que aparece.
3. Crea un nuevo estilo con nombre MiEstilo.Botones. Tiene que modificar el android:padding (tamaño en dp) y android:textSize (el tamaño en sp). Observa que en este proyecto no hay fichero dims.xml (por el tipo de plantilla escogida)
4. Aplícalo a algunos de los botones y visualiza el resultado.

Ejercicio 3: Uso de Temas propios en actividades

5. Usando la herramienta Material Palette, descarga el fichero correspondiente a los 2 colores que hayas preferido. Ábrelo con el editor de texto que prefieras y añade los valores al fichero colors.xml del proyecto P_02_Creciente_01 (realmente solo necesitamos los 2 primarios y el secundario).
6. Crea un tema con nombre MiTema que herede de AppCompat.Light.NoActionBar y que use los colores que has añadido:

```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>

    <style name="MiEstilo">
        <item name="android:textColor">@color/miNaranja</item>
    </style>

    <style name="MiEstilo.Botones">
        <item name="android:padding">25dp</item>
        <item name="android:textSize">30sp</item>
    </style>

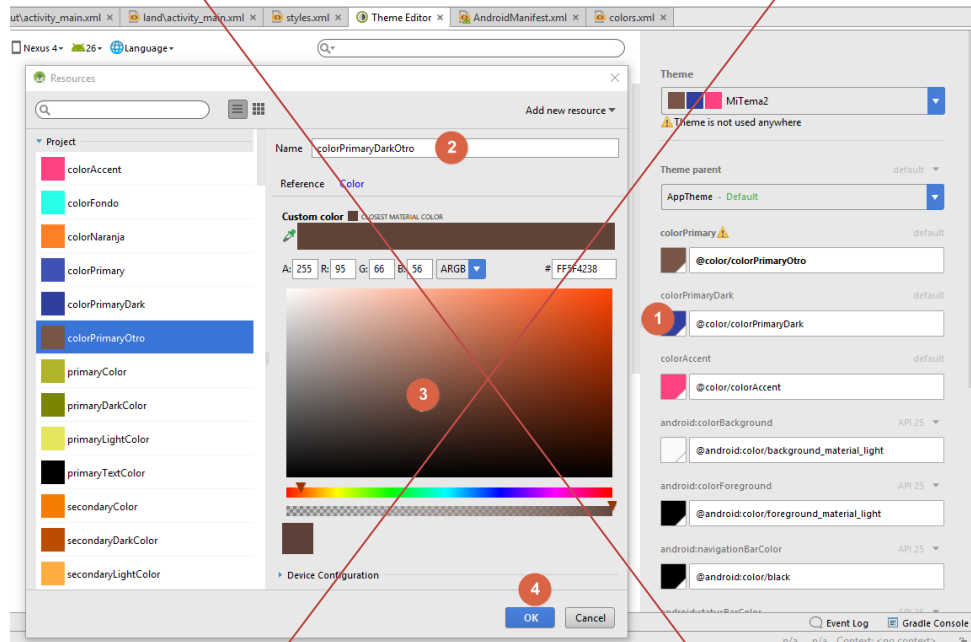
    <style name="MiTema" parent="Theme.AppCompat.Light.NoActionBar">
        <item name="colorPrimary">@color/primary</item>
        <item name="colorPrimaryDark">@color/primary_dark</item>
        <item name="colorAccent">@color/accent</item>
    </style>
</resources>
```

7. Aplica este tema a la actividad principal.
8. Ejecuta en dispositivo.

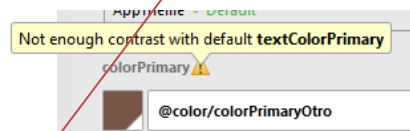
Ejercicio 4: Uso de Temas propios en la aplicación

NO SE PUEDE HACER CON AS 3.5

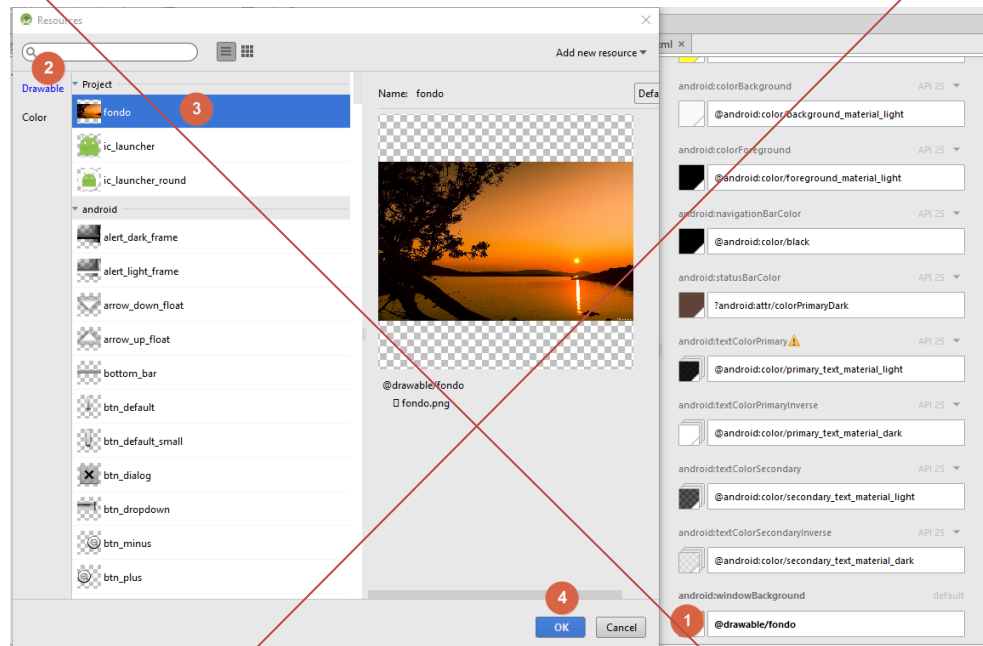
9. Crea un tema con nombre MiTema2 que herede de AppTheme.
10. AS es tan versátil que el Editor de temas hace que sea innecesario el uso de las herramientas mencionadas (son útiles para hacernos una "idea estética" de cómo puede quedar). Cambia la paleta de colores de MiTema2 a una que sea de tu gusto desde el propio Editor de temas, teniendo cuidado de renombrar los colores:



11. Observa que según los colores escogido el Editor de Temas nos advierte con avisos que puede haber poco contraste entre ellos.



- 12.** Copia a la carpeta drawable una imagen de nombre fondo.jpg (o fondo.png) y aplícala como imagen de fondo:



- 13.** Aplica este tema a la aplicación y ejecútala. ¿Por qué no se ve la imagen de fondo? Se nos ha olvidado quitar el tema en la actividad. Hazlo y vuelve a ejecutar.
- 14.** Prueba en dispositivos de distintos tamaños y/o densidades. Se ve bien siempre? No. Recuerda guardar la imagen en distintos tamaños en recursos alternativos de drawables.