Desafío - Ambiente de desarrollo y sus elementos de configuración (II)

En este desafío validaremos nuestros conocimientos sobre los assets de un proyecto Android. Para lograrlo, necesitarás aplicar tus aprendizajes para agregar imágenes vectoriales, modificar el icono de la app usando los iconos de material design con un layout para ordenar las vistas, además de modificar nombres, colores y la utilización de plantillas incluidas en Android Studio.

Lee todo el documento antes de comenzar el desarrollo **en parejas**, para asegurarte de tener el máximo de puntaje y enfocar bien los esfuerzos.

Descripción

Aplicando los conceptos y herramientas aprendidas hasta ahora, crearemos un proyecto y modificaremos parte de su aspecto, utilizando los distintos elementos que componen un proyecto Android.

Estamos trabajando en el inicio de nuestra aplicación, la cual queremos que tenga una pantalla de inicio ocupando toda pantalla (Splash Screen), ya que muchas aplicaciones la utilizan para darle la bienvenida al usuario la primera vez que abre la app.

Por ello, vamos a crear una pantalla de inicio personalizada, ¡donde tú eliges los textos e imágenes a utilizar!

¡Manos a la obra!

Requerimientos

Cumple con los siguientes requerimientos y al finalizar, entrega el **proyecto comprimido en formato zip**:

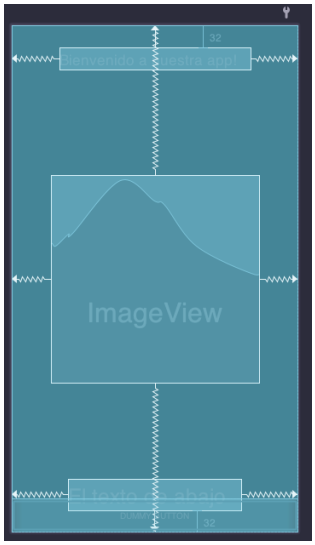
1. Crea un nuevo proyecto usando el template de Fullscreen Activity y asegúrate de que sea compatible con el 100% de los dispositivos. Además, define el nombre de la app para poder crearla.
   (**1 Punto**)

2. Personalizar el icono de la app. Para esto crea un nuevo icono utilizando la biblioteca de iconos de material design y actualiza el *ic_launcher* de la aplicación.
   (**1 Punto**)

3. Abre el archivo activity_fullscreen.xml y reemplaza el TextView con un Constraint Layout indicando `android:id="@+id/fullscreen_content"`. Es importante **mantener el mismo *id***. Revisa las consideraciones para más detalles.
   (**1 Punto**)

4. Con la ayuda de Vector Asset agrega una imagen usando la biblioteca de iconos de material o una imagen local en formato PSD.
   (**1 Punto**)

5. Completa el diseño dentro del ConstraintLayout con 1 TextView en la parte superior, otro en la parte inferior y al centro de la pantalla un ImageView.

   (3 Puntos)

6. Define una dimensión (en el archivo dimens.xml) para utilizar en los márgenes y asigna los márgenes superior e inferior (32 dp).

(**1 Punto**)

7. Actualiza los textos de arriba y abajo, para extraerlos a strings.xml.
(**1 Punto)**

8. Ejecuta en dispositivo y prueba el resultado.
(**1 Punto**)

 ¡Mucho éxito!

Consideraciones y recomendaciones

- Con el layout `activity_fullscreen` abierto, y en modo de diseño, se puede convertir el `TextView` en `ConstraintLayout` directamente desde la sección Component Tree, con un click derecho sobre el `TextView`.

- El ID lo referencia FullscreenActivity en el `onCreate()`, por eso es importante que se mantenga el mismo ID.

```
// Set up the user interaction to manually show or hide the system UI.
            fullscreenContent = binding.fullscreenContent
        fullscreenContent.setOnClickListener { toggle() }
```

- Al usar un `ConstraintLayout`, se necesita actualizar el archivo FullscreenActivity.kt y cambiar el tipo variable definido para **`fullscreenContent`**, de `TextView` a `ConstraintLayout`.

- La línea 24 del archivo FullscreenActivity.kt debería verse así:

```
private lateinit var fullscreenContent: ConstraintLayout
```