Android Styling

David Baeza

01

Estilos

- Un estilo es una colección de propiedades que especifican la apariencia y el formato de una vista.
- Permiten especificar propiedades, como altura, relleno, color de fuente, tamaño de fuente, color de fondo, entre otras.
- Se definen en ficheros XML dentro del directorio *res/values* de un proyecto de Android Studio.

01

Uso de estilos

```
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textColor="#00FF00"
    android:typeface="monospace"
    android:text="@string/hello" />
```

Vista sin estilos

```
<TextView
    style="@style/CodeFont"
    android:text="@string/hello" />
```

Vista con estilos

01

Definición de estilos

name: *nombre que identifica exclusivamente al estilo*

parent: *nombre del estilo incorporado de Android del cual se va a heredar*

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="CodeFont" parent="@android:style/TextAppearance.Medium">
        <item name="android:layout_width">fill_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textColor">#00FF00</item>
        <item name="android:typeface">monospace</item>
    </style>
</resources>
```

<item>: *propiedad del estilo que se va a especificar*

01

Herencia de estilos

- Usando el atributo *parent* del elemento <*style*>, es posible proporcionar propiedades heredables a un estilo.

- Para heredar estilos propios, no se debe usar el atributo *parent.* En su lugar, solo se debe indicar el nombre del estilo del que se desea heredar, como prefijo del nombre del estilo nuevo (separados por un punto).

  Ejemplo: NombreEstiloPropio.NombreEstiloNuevo

01

Herencia de estilos

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="CodeFont" parent="@android:style/TextAppearance.Medium">
        <item name="android:layout_width">fill_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textColor">#00FF00</item>
        <item name="android:typeface">monospace</item>
    </style>
</resources>
```

Definición de estilo propio

```xml
    <style name="CodeFont.Red">
        <item name="android:textColor">#FF0000</item>
    </style>
```

Herencia de propiedades de estilos propios

01

Herencia de estilos

```
<style name="GreenText" parent="@android:style/TextAppearance">
    <item name="android:textColor">#00FF00</item>
</style>
```

Herencia de propiedades de estilos incorporados en Android

02

Temas

- Un tema es un estilo que se aplica a toda una Activity o aplicación, y no a una vista individual.
- Cuando un estilo se usa como un tema, cada vista de la actividad o aplicación usará todas las propiedades de estilo que admite.
- Los temas se aplican dentro de las etiquetas <application> o <activity> en el archivo AndroidManifest.xml, especificando en el atributo android:theme, el nombre del estilo previamente creado.

02

Uso de temas

```
<application android:theme="@style/CustomTheme">
```

Definición de un tema para una aplicación entera

03 Layouts

- Definen la estructura de una interfaz de usuario dentro de una aplicación.
- Los layouts más comunes integrados en Android son: *Linear Layout, Relative Layout, Web View, Constraint Layout.*
- También es posible crear layouts a través de adaptadores: *List View, Grid View.*

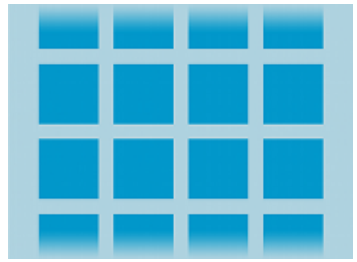
03 Layouts

Linear Layout

Relative Layout

Web View

List View



Grid View

03

Constraint Layout

- Permite crear diseños grandes y complejos con una jerarquía de vistas plana (sin grupos de vistas anidadas).
- Es similar a *Relative Layout* ya que se presentan todas las vistas relacionadas entre sí. La diferencia radica en la flexibilidad y las herramientas visuales que el editor de diseño del *Constraint Layout* ofrece.
- El concepto de *Constraint Layout* está fundamentado en la idea de crear interfaces de usuario, arrastrando y soltando elementos.