Interfaz de usuario en Android: CardView

Junto a Android 5.0 Lollipop nos llegó un nuevo componente que, si bien se llevaba utilizando ya algún tiempo en diversas aplicaciones, no tenía soporte directo en el SDK. Este nuevo componente llamado `CardView` es la implementación que nos proporciona Google del elemento visual en forma de tarjetas de información que tanto utiliza en muchas de sus aplicaciones, entre ellas Google Now, quizá la que más a ayudado a popularizar este componente.

Hasta la llegada de Android 5.0, para utilizar estas "tarjetas" en la interfaz de nuestras aplicaciones teníamos que recurrir a librerías de terceros o bien trabajar un poco para implementar nuestra propia versión. Sin embargo ahora las tenemos disponibles en forma de nueva librería de soporte oficial, proporcionada junto al SDK de Android.

Para hacer uso de la librería tan sólo tendremos que hacer referencia a ella en la sección de dependencias del fichero*build.gradle* del módulo principal de la aplicación:

```
1    dependencies {
2        ...
3        compile 'com.android.support:cardview-v7:21.0.+'
4    }
```

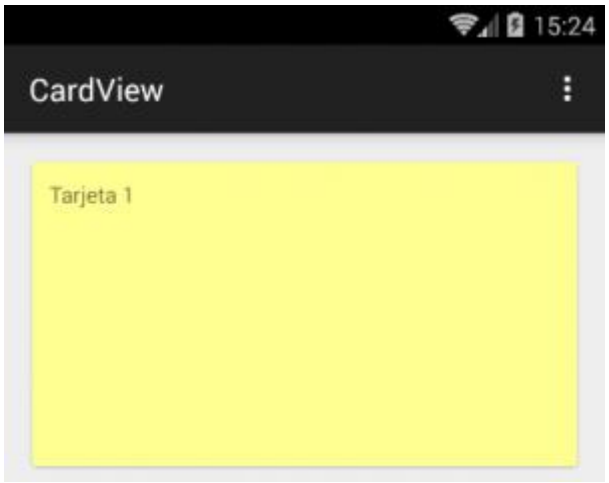
Una vez incluida la referencia a la librería, la utilización del componente es muy sencilla. Tan sólo tendremos que añadir a nuestro layout XML un control de tipo `<android.support.v7.widget.CardView>` y establecer algunas de sus propiedades principales. Yo por ejemplo le asignaré una altura de 200dp (`layout_height`), una anchura que se ajuste a su control padre (`layout_width`), y un color de fondo amarillo estilo post-it (`cardBackgroundColor`).

```
1        <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
             xmlns:card_view="http://schemas.android.com/apk/res-auto"
```

```
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:orientation="vertical"
        tools:context=".MainActivity">

        <android.support.v7.widget.CardView
            android:id="@+id/card2"
            android:layout_width="match_parent"
            android:layout_height="200dp"
            card_view:cardBackgroundColor="#fffffe91" >

            <TextView
                android:id="@+id/txt1"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:padding="10dp"
                android:text="@string/tarjeta_1" />

        </android.support.v7.widget.CardView>

    </LinearLayout>
```

Por supuesto, como en el caso de cualquier otro *contenedor* (un `CardView` no es más que una extensión de `FrameLayout` con esquinas redondeadas y una sombra inferior), dentro de un `CardView` podemos añadir todos los controles que necesitemos. Como podéis ver en el código anterior, a modo de ejemplo he añadido tan solo una etiqueta de texto (`TextView`).

Si ejecutamos en este momento el ejemplo sobre un dispositivo o emulador el resultado que esperábamos:



Es posible que existan pequeñas diferencias entre la vista en dispositivos (las sombras del cardView) Android 5.x o superiores y dispositivos de versiones anteriores, para solucionarlo se añade lo siguiente.

```
1
2
3
4
5
6
```

```
<android.support.v7.widget.CardView
    android:id="@+id/card2"
    android:layout_width="match_parent"
    android:layout_height="200dp"
    card_view:cardBackgroundColor="#fffffe91"
    card_view:cardUseCompatPadding="true" >
```

Además del color de fondo que ya hemos comentado, también podremos definir la elevación de la tarjeta y el radio de las esquinas redondeadas, utilizando las propiedades `cardElevation` y `cardCornerRadius` respectivamente.

No son muchas más las opciones de personalización que nos ofrece `CardView`, pero con poco esfuerzo deben ser suficientes para crear tarjetas más "sofisticadas", jugando por supuesto también con el contenido de la tarjeta. Como ejemplo, si incluimos una imagen a modo de fondo (`ImageView`), y una etiqueta de texto superpuesta y alineada al borde inferior (`layout_gravity="bottom"`) con fondo negro algo traslúcido (por ejemplo`background="#8c000000"`) y un color y tamaño de texto adecuados, podríamos conseguir una tarjeta con el aspecto siguiente en **Android 5.x**:

El código concreto para conseguir lo anterior sería el siguiente:

```
1
2
3       <android.support.v7.widget.CardView
4           android:id="@+id/card1"
5           android:layout_width="match_parent"
6           android:layout_height="200dp"
7           card_view:cardCornerRadius="6dp"
8           card_view:cardElevation="10dp"
9           card_view:cardUseCompatPadding="true" >
10
11          <ImageView
12              android:layout_width="match_parent"
13              android:layout_height="match_parent"
14              android:src="@drawable/city"
15              android:scaleType="centerCrop"/>
16
17          <TextView
18              android:id="@+id/txt2"
19              android:layout_width="match_parent"
20              android:layout_height="wrap_content"
21              android:padding="10dp"
22              android:text="@string/tarjeta_2"
23              android:layout_gravity="bottom"
24              android:background="#8c000000"
25              android:textColor="#ffe3e3e3"
26              android:textSize="30sp"
27              android:textStyle="bold"/>

      </android.support.v7.widget.CardView>
```

Todo parece correcto, pero si ejecutamos el ejemplo en un dispositivo/emulador con **Android 4.x** veremos lo siguiente:

Para evitar este margen adicional en Android 4 y anteriores, puede asignarse el valor `false` a la propiedad `cardPreventCornerOverlap` del `CardView`.

```
    <android.support.v7.widget.CardView
        android:id="@+id/card1"
        android:layout_width="match_parent"
        android:layout_height="200dp"
        card_view:cardCornerRadius="6dp"
        card_view:cardElevation="10dp"
        card_view:cardUseCompatPadding="true"
        card_view:cardPreventCornerOverlap="false" >
```

Esto evitará que se incluya el margen extra alrededor de la imagen, aunque no redondeará las esquinas de la imagen, quedando un efecto extraño respecto a la sombra del `CardView` si se ha utilizado un radio amplio para redondear las esquinas (en el ejemplo he utilizado un radio alto para que se note bien el efecto, si el radio es menor podría pasar más desapercibido).

Puedes consultar y/o descargar el código completo de los ejemplos desarrollados en este artículo accediendo a la página del curso en GitHub.