4 – LAYAOUTS y VISTAS Avanzadas

Material Design

Material Design es un conjunto de pautas de diseño, iconos y objetos visuales que le dan soporte al concepto, en un intento de Google de renovar y estandarizar el aspecto visual de las aplicaciones (ya sean web, móviles o para otros dispositivos)

Material Design

Material Design pone énfasis en:

Objetos limpios, bien definidos

Jugar con las sombras y luces

Predefinir animaciones (movimientos)

Design Support Library

Es la librería de Android que implementa las guías de Material Design y permite su uso en versiones previas a la versión 5 (Lolipop) -backward-

Para su uso, debemos importarla en el fichero de gradle del módulo (app)

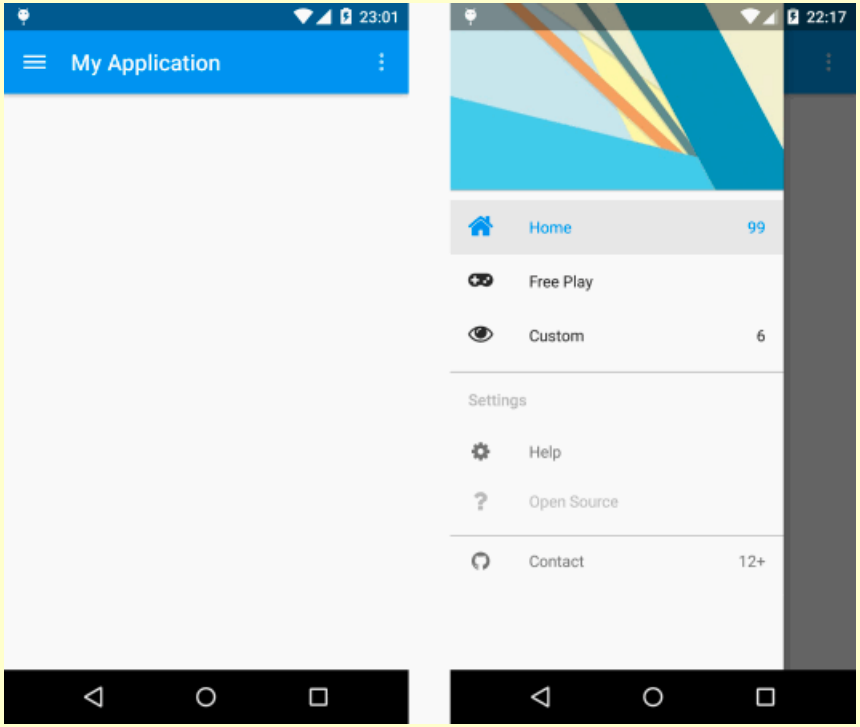
Design Support Library

Para poder usar esa librería es necesario importarla explícitamente en el fichero de gradle del módulo (app)

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:23.3.0'
    compile 'com.android.support:design:23.3.0'
}
```

NavigationView

El menú estándar de navegación



NavigationView

```
<android.support.v4.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"  android:layout_width="match_parent"
    android:layout_height="match_parent"  android:fitsSystemWindows="true">
    <!-- referencia a layout activity main -->
    <android.support.design.widget.NavigationView
        android:id="@+id/navigation" android:layout_width="wrap_content"
        android:layout_height="match_parent"  android:layout_gravity="start"
        app:menu="@menu/my_navigation_items" />
 </android.support.v4.widget.DrawerLayout>
```

NavigationView

El DrawerLayout (elemento Raíz) alberga:

- El Layout de la actividad (activity_layout)
- El menú lateral (NavigationView)

NavigationView

Subelementos importantes

android:layout_gravity="start"

app:headerLayout="@layout/cabecera"

app:menu="@menu/menu_navegacion"

NavigationView

Código relevante

getSupportActionBar()

setHomeAsUpIndicator(ID-imangen)

setDisplayHomeAsUpEnabled(true);

NavigationView Menu

<menu> Para definir secciones

<item> Defino las opciones en sí

<group> Agrupo items, pudiendo hacer que se muestren o desparezcan, aparezcan activados o no, como conjunto

NavigationView Menu

Como atributo destacado de item puedo encontrar android:showAsAction, aplicado al appBar:

Always: Siempre se muestra en bar

IfRoom : Si cabe, se muestra

Never: Nunca se muestra en la bar

WithText: Se muestra con imagen y texto

CollapseActionView: Se muestra oculto

Referencia a un layaout

Puedo modular la descripción de layouts y segmentarla en diferentes ficheros a fin de tener más claridad en mis diseños con el elemento include

<include layout="@layout/contenido_layout" />

Referencia a estilos

Para emplear un color o una medida del tema en uso, puedo usar una referencia a ese atributo, en vez de poner un valor explícito.

Ello se hace con ?nombreatributo ó

?([android:] [app:])*attr/?nombreatributo

*(Theme SDK) ó definido en la propia app

Iconos Material Desing

Aparte de definir nuestros propios iconos, Google nos ofrece a través de su web , multitud de iconos en diferentes resolociones listos para ser incluidos como recursos de nuestra aplicación, de descarga gratuita

https://design.google.com/icons/

Iconos Material Desing

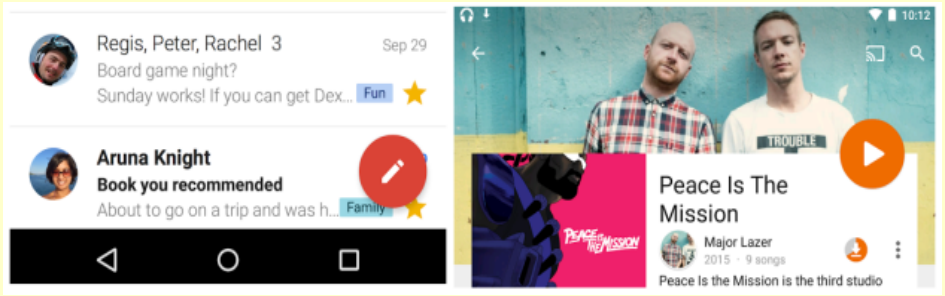
Los iconos, podemos encontrarlos como subcarpeta de res/drawable o res/mipmap

En esencia, es lo mismo, aunque mipmap está hecho con la intención de albergar los iconos de lanzamiento de la app (launcher)

Floating Action Button

FAB es es un botón que flota por encima del Layout. Por defecto, se muestra en la zona derecha.

Su función es lanzar una actividad común



Floating Action Button

Este botón también se puede transformar (morphing) en:

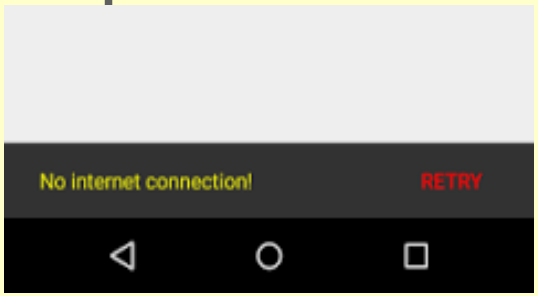
- Un menú contextual
- Una barra

Es un comportamiento No estándar

SnackBar

Pequeño cuadro de diálogo previsto para mostrar un mensaje después de una operación

Desaparece al poco automáticamente



TextInputLayout

Layout que envuelve a un elemento TextView y le aporta propiedades dinámicas que mejoran su usabilidad y diseño de cara al usuario

TextInputLayout

```
<android.support.design.widget.TextInputLayout
android:layout_width="match_parent"
android:layout_height="wrap_content">

<android.support.design.widget.TextInputEditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="@string/form_username"/>

</android.support.design.widget.TextInputLayout>
```

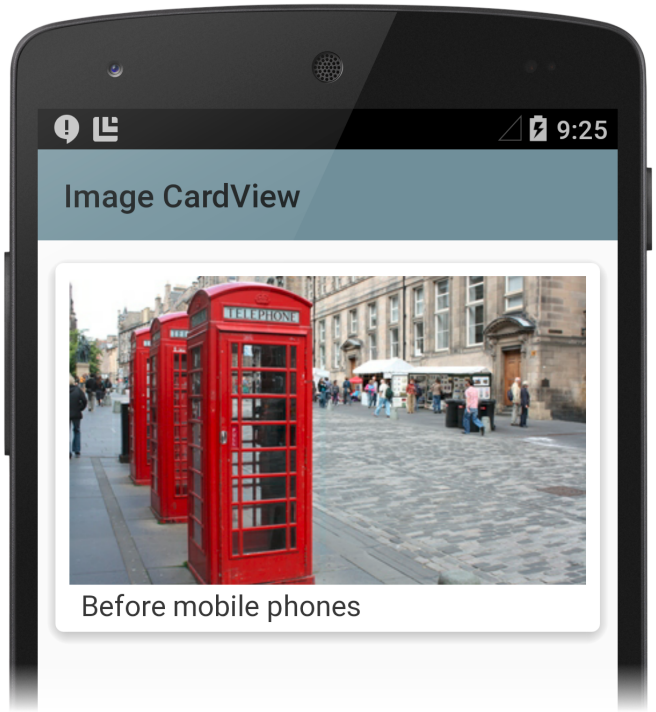
CardView

Un CardView es un FrameLayout pero con esquinas redondeadas, sombras y fondo

Fue añadido en la versión 5 por lo que es necesario incluir la librería que garantice compatiblidad hacia atrás

compile 'com.android.support:cardview-v7:23.3.0'

CardView

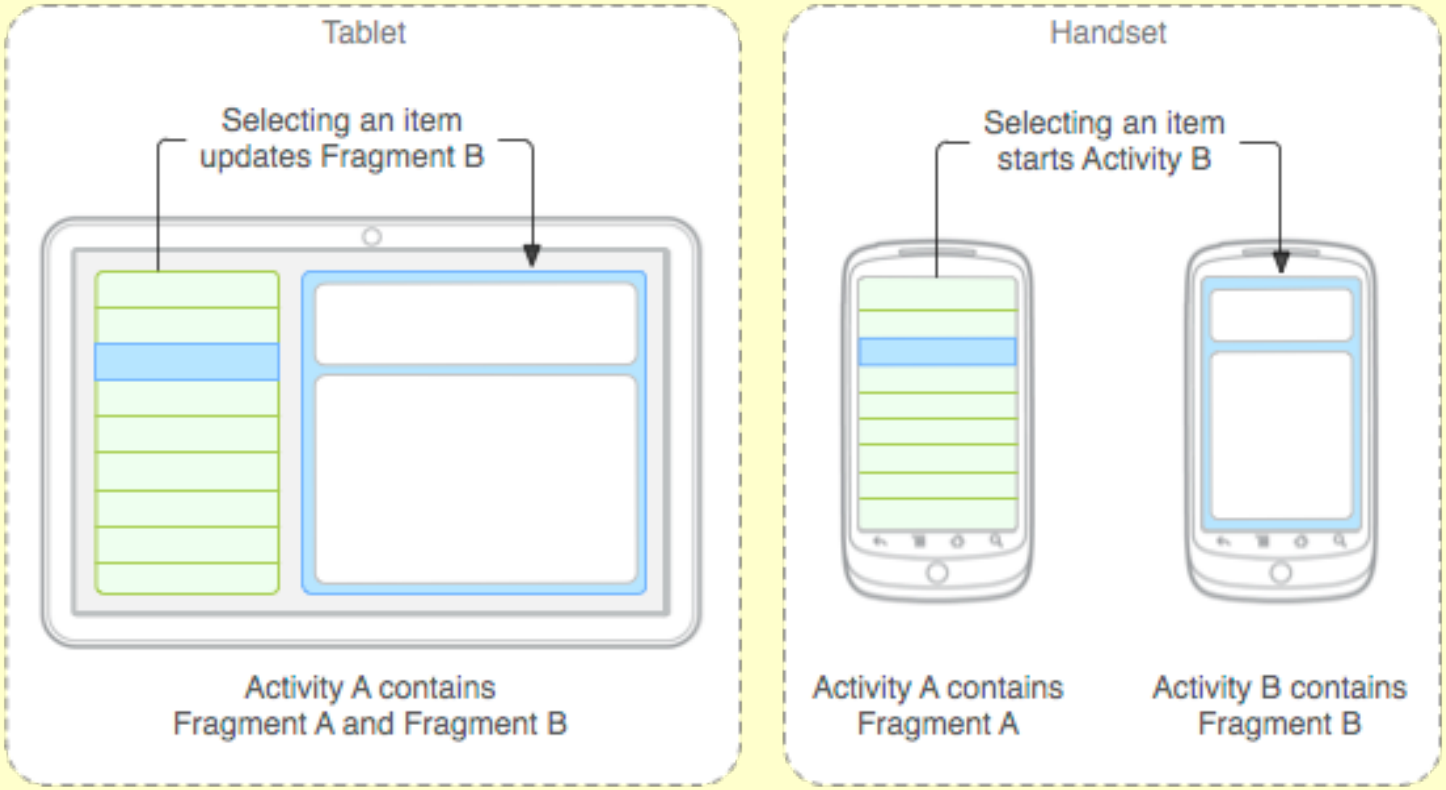


Fragment

Un fragment es una parte de la interfaz de usuario. Es un “fragmento” de pantalla, que puede ser reutilizado en múltiples actividades.

- Siempre pertenece a una Activity
- Tiene su propio ciclo de vida
- Gestiona sus propios eventos

Fragment


Tablet
Selecting an item updates Fragment B
Activity A contains Fragment A and Fragment B
Handset
Selecting an item starts Activity B
Activity A contains Fragment A
Activity B contains Fragment B

Fragment

Para crear un fragment debo crear una subclase de Fragment e implementar estos métodos:

- onCreate → Para inicializar/recrear componentes
- onCreateView* → Dibuja y devuelve la View raíz
- onPause → Para guardar el estado

*obligatorio

Fragment

Puedo declarar un Fragment por código como nodo en el Layout de su Activity o en tiempo de ejecución

```
<fragment
android:name="com.example.ArticleListFragment"
        android:id="@+id/list"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="match_parent" />
```

Fragment

Para instanciar los Fragments desde el código,debo usar la clase FragmentManager desde la Activty de turno.

Para abrir, cerrar y saltar de un Fragment a otro, se emplea FragmentTransaction

Fragment

Para compartir info entre fragments, es necesario que esta pase por la Activity contenedora.

Para ellos, se hace un pequeño artificio:

Fragment

1 Defino una interfaz como atributo del Fragment

2 Hago que la Activity contenedora implemente esa interfaz

3 Asocio la Activity, como listener de los elementos del Fragment

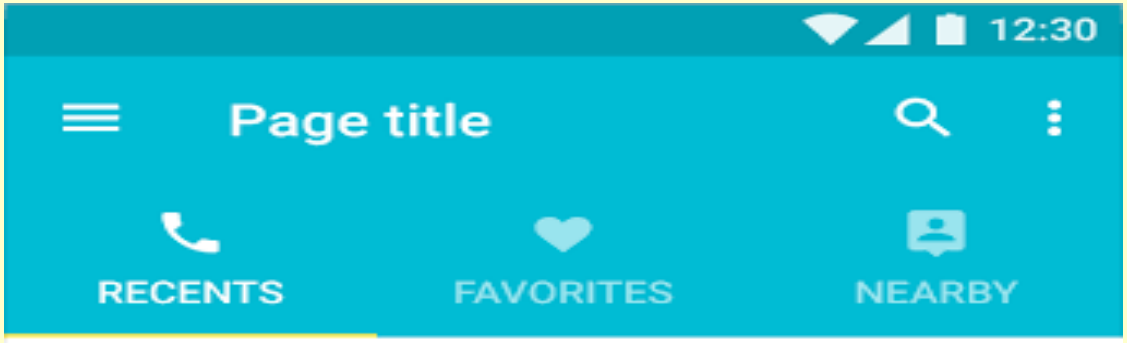
ViewPager

Es una clase que permite transitar automáticamente entre pantallas. Cada pantalla será un Fragment

ViewPager empleará un PagerAdapter como proveedor para obtener cada Fragment.

TabLayout

Podemos crear una fila horizontal de selectores que aparezcan a modo de menú horizontal fácilmente gracias a TabLayout



TabLayout

```
android.support.design.widget.TabLayout
        android:id="@+id/tabLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="?attr/colorPrimary"
        android:minHeight="?attr/actionBarSize"

android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"/>
```

TabLayout

Para su inicialización, obtengo referencia a él desde el código y relleno sus elementos

```
TabLayout tabLayout = findViewByID (R.id.tabLayout);
tabLayout.addTab(tabLayout.newTab().setText("Tab 1"));
tabLayout.addTab(tabLayout.newTab().setText("Tab 2"));
tabLayout.addTab(tabLayout.newTab().setText("Tab 3"));
```

TabLayout

Aunque también puedo añadir Items desde el XML

```
<android.support.design.widget.TabItem
        android:text="@string/tab_text"/>

    <android.support.design.widget.TabItem
        android:icon="@drawable/ic_android"/>
```

TabLayout

Debo completar el funcionamiento con un Listener que se invoque ante los distintos eventos sobre el TabLayout

onTabSelected(TabLayout.Tab tab)

onTabUnselected(TabLayout.Tab tab)

onTabReselected(TabLayout.Tab tab)

RecyclerView

Es la evolución de ListView y GridView. Debemos definir:

ADAPTER (datos)

VIEWHOLDER (referencia a contenido)

LAYOUTMANAGER (cómo se muestra)

RecyclerView

De forma opcional:

ITEMDECORATION (separación vistas)

ITEMANIMATOR (modificación elementos)