Programación de aplicaciones Android

Dia 2

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

Fabián Zamora Ramírez
Jeff Schmidt Peralta

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

I. Estilos y temas

Un **estilo** es una colección de propiedades que especifican la apariencia y el formato de un view o ventana. Un estilo puede especificar propiedades, como altura, relleno, color de fuente, tamaño de fuente, color de fondo y mucho más. Los estilos se definen en un recurso xml que está separado del xml que especifica el diseño.

Un **tema** es un estilo que se aplica a toda una Activity o aplicación, y no a una view individual. Cuando un estilo se usa como un tema, cada vista de la actividad o aplicación usará todas las propiedades de estilo que admite.

A. Temas para versiones menores a Android 5.0

Android provee temas estándar para mantener mayor cohesión entre aplicaciones. En versiones de Android a partir de 3.0 y hasta 5.0 (exclusive) se incluyen 3 temas principales que se pueden utilizar:

- Holo Light
- Holo Dark
- Holo Light con Action Bar oscuro

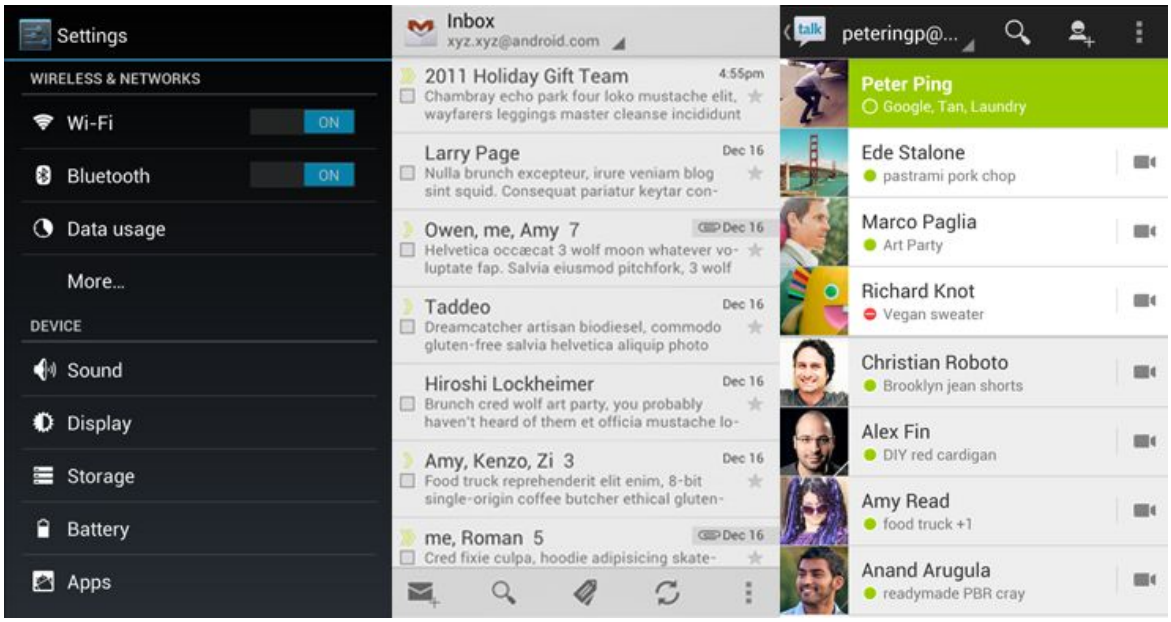

Figura 1. De izquierda a derecha: Holo, Holo Light & Holo Light con Action Bar oscuro

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

B. Temas para versiones mayores a Android 5.0

Google introduce en 2014 Material Design. Material es una guía para el diseño visual, de movimientos y de interacción en distintas plataformas y dispositivos. Esta se integró en Android Lollipop como reemplazo de Holo.

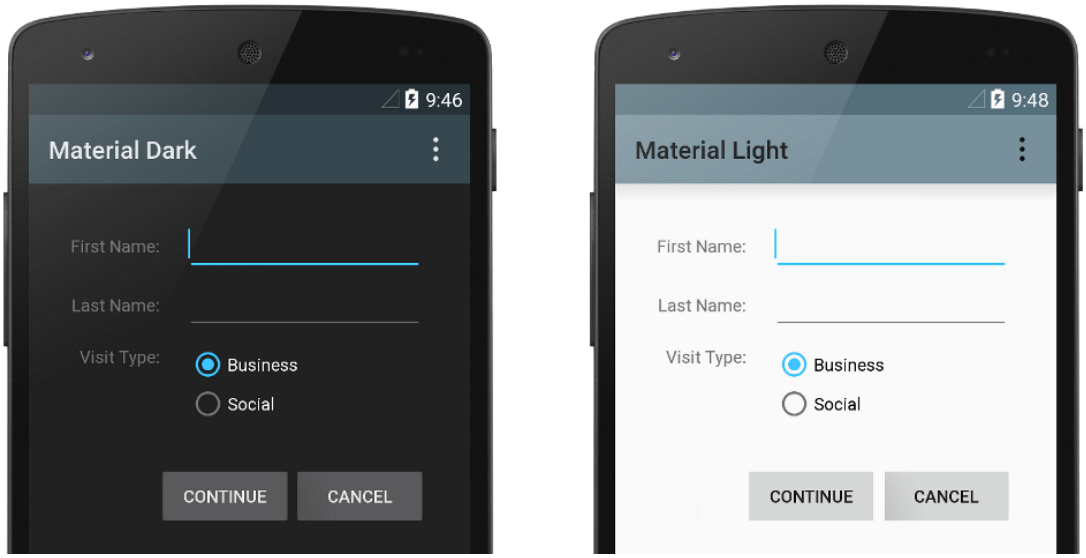

Figura 2. De izquierda a derecha: Material Dark & Material Light

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

II. Material design

Como se mencionó anteriormente, Material Design es una guía para el diseño visual, de movimientos y de interacción en distintas plataformas y dispositivos introducida por Google. Esta guía no abarca solo Android, sino que también se ha incorporado tanto a web como a otras plataformas móviles como iOS.

En Android, para crear aplicaciones con Material Design se debe:

- Revisar la especificación de Material Design que se encuentra disponible en https://material.google.com.
- Aplicar el tema material a la aplicación.
- Seguir las pautas de Material Design.
- Especificar la elevación de las vistas para convertir sombras.
- Usar widgets del sistema para listas y tarjetas.
- Personalizar las animaciones de la aplicación.

A. Estructura básica

Material design provee un estructura base para las aplicaciones móviles como se muestra en la Figura 3. Esta estructura incluye una App bar permanente y un botón de acción flotante. Se puede añadir una barra inferior opcional para funcionalidad adicional. Los menús de navegación lateral superponen todos los demás elementos estructurales.

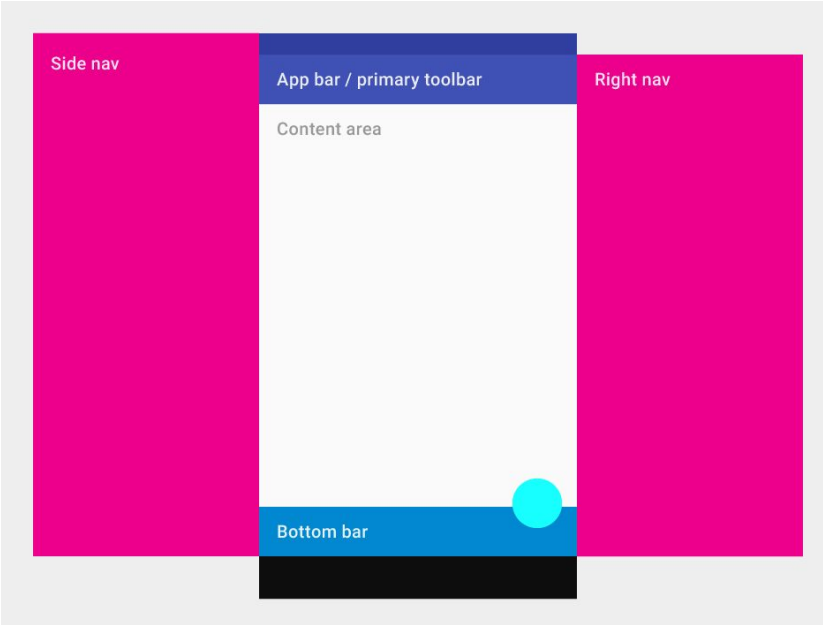

Figura 3.Estructura básica para una aplicación utilizando Material Design.

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

Toolbars

Representan una barra de tareas dentro de la aplicación. Estas tienen muchas funcionalidades. Algunos ejemplos se muestran en las Figuras 4, 5 y 6.

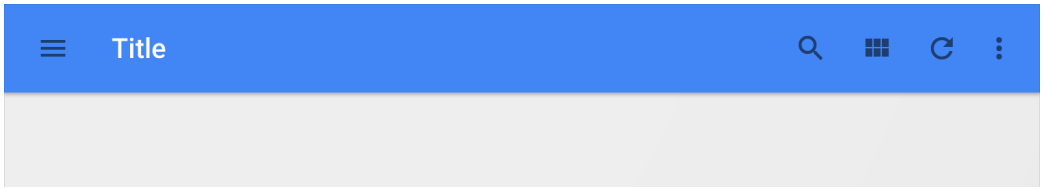

Figura 4. Toolbar de tamaño estándar para un App bar.

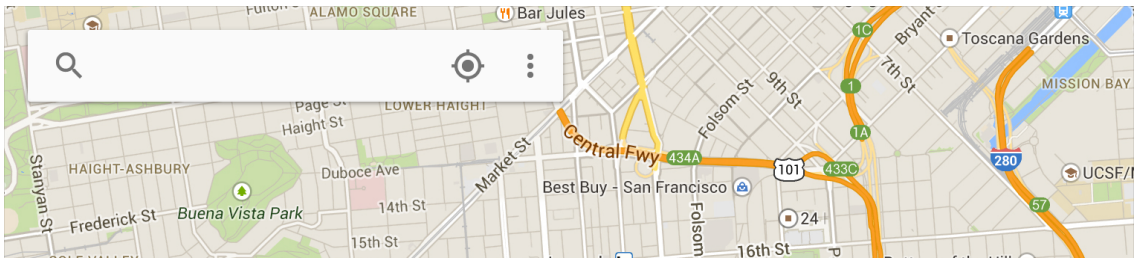

Figura 5. Toolbar flotante.

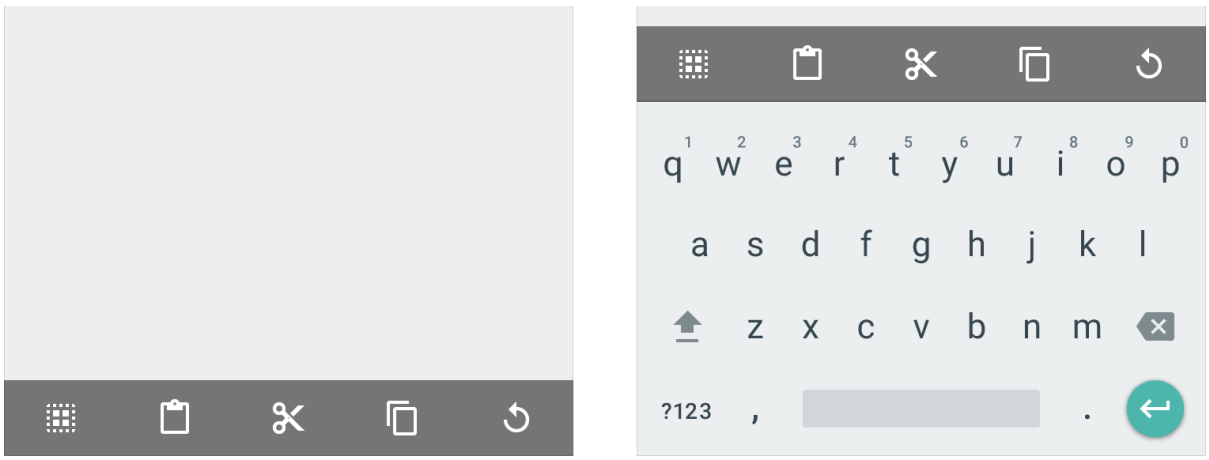

Figura 6. Toolbar inferior que se acomoda según el teclado.

App Bar

El **app bar**, anteriormente conocida como **action bar** en Android, es un tipo de **toolbar** que se utiliza para resaltar la marca, navegación, búsqueda y acciones.

El icono de navegación en el lado izquierdo de la barra de aplicaciones puede ser:

- Un control para abrir un menu de navegación.

- Una flecha hacia atrás para navegar hacia atrás a través de la jerarquía de la aplicación.
- Se omite si no se requiere ninguna navegación desde esta pantalla.

El título de la barra de aplicaciones refleja la página actual. Puede ser un título de aplicación, un título de página o un filtro de página. Los iconos en el lado derecho de la barra de aplicaciones son acciones relacionadas con la página actual. El icono de menú abre el menú de acciones extras, que contiene acciones secundarias y elementos de menú como ayuda, ajustes y retroalimentación.

A partir de Android 3.0 (API 11), todas las actividades que utilizan el tema predeterminado tienen un **action bar**. Sin embargo, nuevas funcionalidades se han ido añadiendo gradualmente a esta en varias versiones de Android. Como resultado, la **action bar** nativa se comporta de manera diferente dependiendo de la versión del sistema Android que esté utilizando un dispositivo. Como solución, ahora las características más recientes se añaden a la versión de la **action bar** de la biblioteca de soporte, **Toolbar**, y están disponibles en cualquier dispositivo que pueda utilizar la biblioteca de soporte.

El uso de la barra de herramientas de la biblioteca de soporte (**Toolbar**) ayuda a garantizar que la aplicación tenga un comportamiento uniforme en la más amplia gama de dispositivos.

Figura 7. Estructura del App bar.

Menús

Un menu es como una hoja de papel que se muestra sobre el app bar y muestra acciones extra.

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

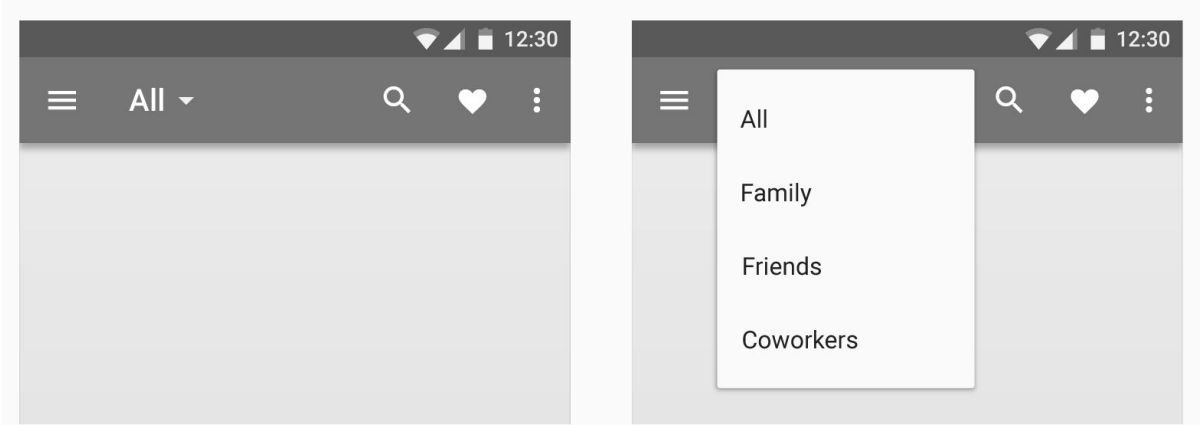

Figura 8. Ejemplo de un menu en el app bar.

Barra de navegación de Android

La barra de navegación de Android contiene los controles de navegación del dispositivo: Atrás, Inicio y Descripción general. También muestra un menú para aplicaciones escritas para Android 2.3 o versiones anteriores.

Figura 9. Barra de navegación de Android.

Barras de navegación laterales

Si están presentes, las barras de navegación laterales pueden ser mostradas permanente o temporalmente y se pueden ubicar del lado izquierdo o derecho de la pantalla.
El contenido de la barra izquierda debe ser idealmente de navegación. El contenido de la barra derecha debe ser contenido secundario del contenido principal de la ventana.

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

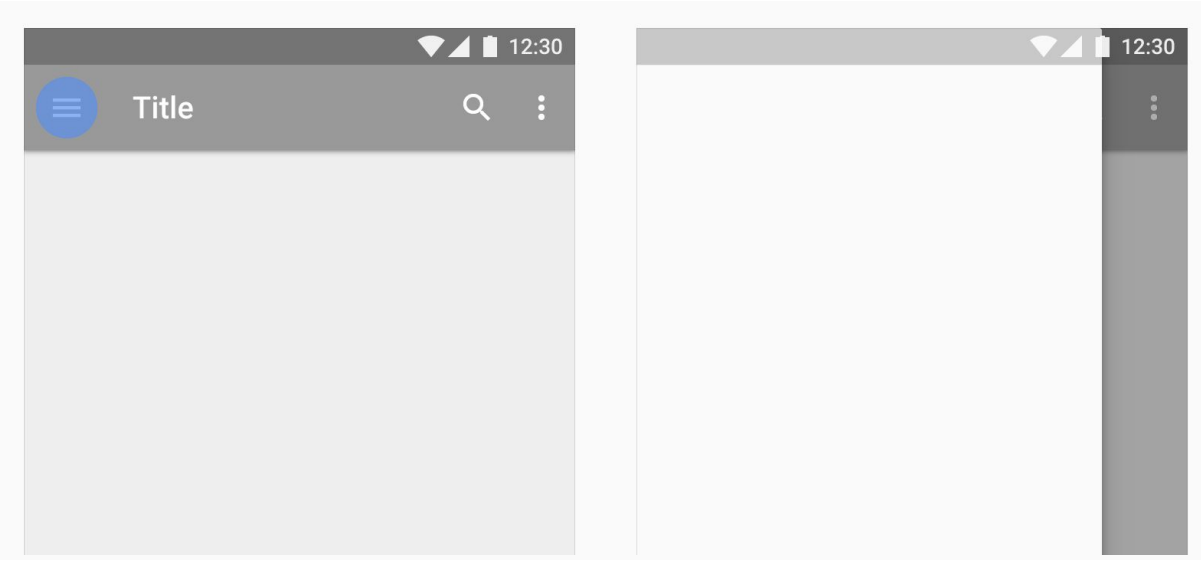

Figura 10. Barra de navegación lateral.

B. Implementación

Algunas características de Material Design, como el tema material (**Theme.Material**) y las transiciones de actividades personalizadas, solo están disponibles en Android 5.0 (API 21) y superior. Sin embargo, se puede lograr que las aplicaciones usen estas características cuando se ejecutan en dispositivos que tienen versiones anteriores de Android mediante el uso de las bibliotecas de soporte **v7-appcompat**.

A continuación se presentan los elementos de la biblioteca de soporte que permiten implementar una estructura básica como la mostrada en la Figura 3.

Tema Material

El biblioteca de soporte **v7-appcompat** ofrece un nuevo estilo para las aplicaciones, widgets del sistema que permiten configurar la paleta de colores y animaciones predeterminadas para información táctil y transiciones de actividades: ***Theme.AppCompat***. Este tema se puede utilizar como base para nuevos temas específicos de las aplicaciones. Por ejemplo, es muy sencillo cambiar los colores base de la aplicación mostrados en la Figura 11, tal como se muestra en la Figura 12.

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

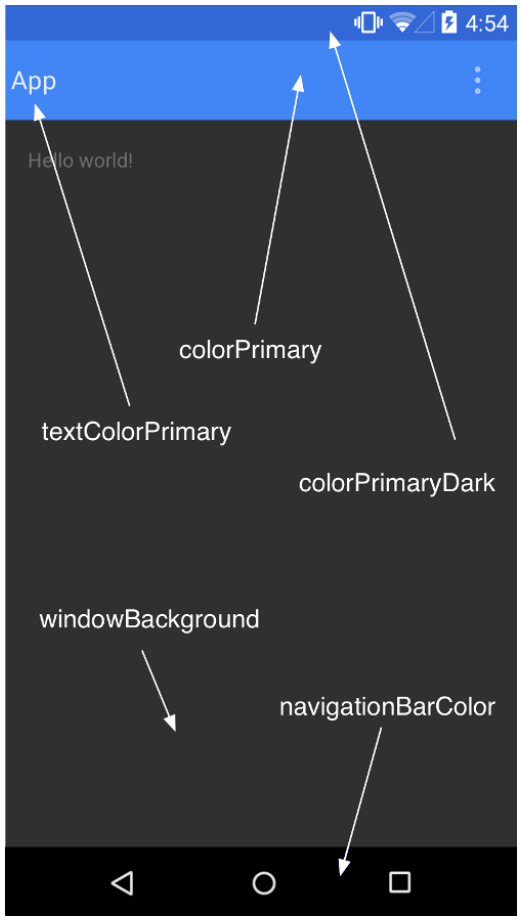

Figura 11. Nombres de los colores predefinidos de una ventana.

```
<!-- 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>
```

Figura 12. Ejemplo de cómo personalizar el tema material.

CoordinatorLayout

Un **CoordinatorLayout** es un **FrameLayout** con superpoderes. Está desarrollado para dos usos principales:

1. Como contenedor principal de una aplicación.
2. Como un contenedor que permite a los view hijos interactuar entre ellos.

Fabián Zamora Ramírez
Jeff Schmidt Peralta

AppBarLayout

Es un **LinearLayout** vertical que implementa muchas de las características del **AppBar** de Material Design, principalmente sobre gestos y scrolling. Este view debe ser utilizado como un hijo directo de un **CoordinatorLayout** para que funcione correctamente.

DrawerLayout

Actúa como un contenedor principal para una ventana que permite menús laterales de uno o ambos lados de la ventana.

NavigationView

Representa un menu de navegación estándar para una aplicación. El menu puede ser rellenado desde un xml. Por lo general se coloca dentro de un DrawerLayout.

ActionBarDrawerToggle

Esta clase proporciona una manera práctica de **unir la funcionalidad** del **DrawerLayout** y el **ActionBar** para implementar el diseño recomendado para el menu de navegación según Material Design.

Para usar ActionBarDrawerToggle, se debe crear uno en la Activity y llamarlo a través de los siguientes métodos correspondientes a la Activity:

- onConfigurationChanged()
- onOptionsItemSelected()

El método ActionBarDrawerToggle#syncState() se debe llamar desde el método Activity#onPostCreate() para sincronizar el indicador con el estado del DrawerLayout vinculado.

FloatingActionButton

Un tipo especial de botón que se distingue por un icono circular flotando por encima de la interfaz de usuario y tienen comportamientos especiales de movimiento.

Uniendo los elementos

La relación entre los elementos mencionados anteriormente se muestra en la Figura 13. La jerarquía de elementos básica es:

- DrawerLayout
    - CoordinatorLayout
        - AppBarLayout
            - Toolbar
            - FrameLayout [Contenedor para Fragments]

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

- NavigationView

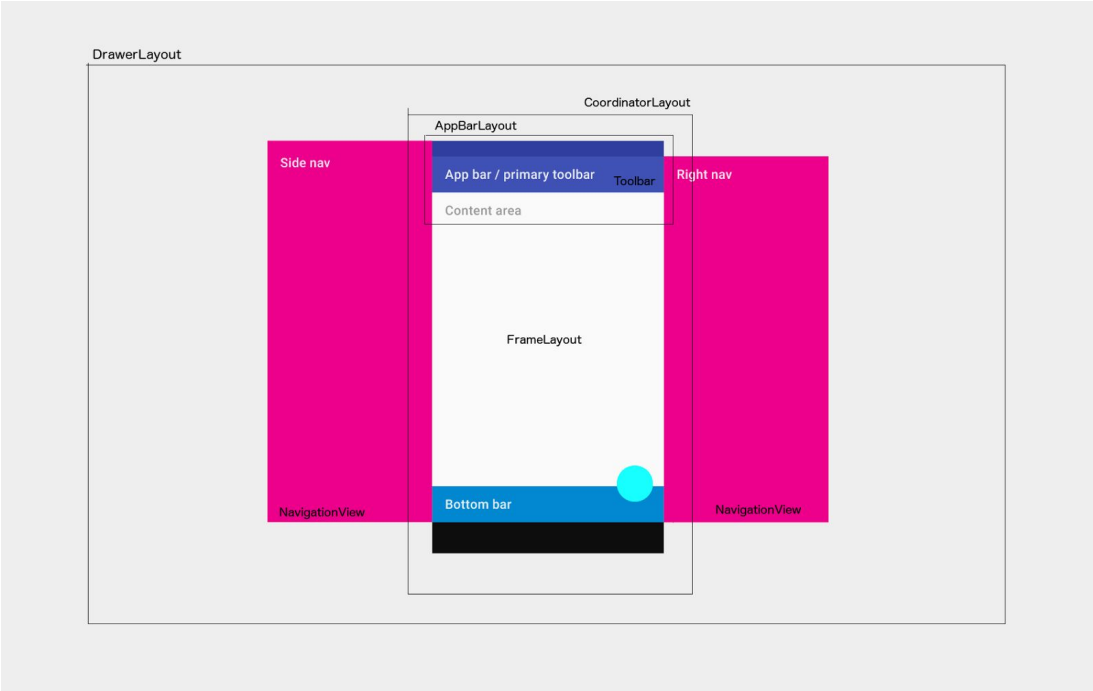

Figura 13. Elementos para implementar la estructura básica de una aplicación Material Design.

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

III. Menús

Los menús son un componente común de la interfaz de usuario en muchos tipos de aplicaciones. Para proporcionar una experiencia de usuario conocida y uniforme, se deben usar el API de **Menu** de Android para presentar al usuario acciones y otras opciones en las actividades.

Para crear un **Menu** se debe declarar un xml dentro de la carpeta *res/menu*, como se muestra en la Figura 14.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/new_game"
          android:icon="@drawable/ic_new_game"
          android:title="@string/new_game"
          android:showAsAction="ifRoom"/>
    <item android:id="@+id/help"
          android:icon="@drawable/ic_help"
          android:title="@string/help" />
</menu>
```

Figura 14. Ejemplo de declaración de un Menu.

Fabián Zamora Ramírez
Jeff Schmidt Peralta
Noviembre, 2016

IV. Referencias

- *Portions of this page are reproduced from work created and shared by the Android Open Source Project and used according to terms described in the Creative Commons 2.5 Attribution License.*
- *Portions of this page are modifications based on work created and shared by the Android Open Source Project and used according to terms described in the Creative Commons 2.5 Attribution License.*
- *La lista completa de sitios se enlista a continuación:*
    - *https://developer.android.com*
    - https://developer.android.com/training/material/get-started.html
    - https://developer.android.com/design/material/index.html?hl=es
    - https://developer.android.com/training/material/theme.html?hl=es
    - *https://developer.android.com/training/material/lists-cards.html?hl=es*
    - *https://developer.android.com/reference/android/app/Application.html*
    - *https://developer.android.com/reference/android/support/v7/widget/RecyclerView.ViewHolder.html*