

INDICE

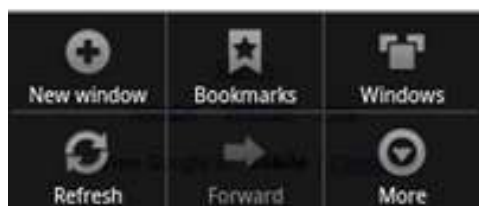
<u>1.</u>	<u>INTRODUCCIÓN</u>	<u>1</u>
<u>2.</u>	<u>MENÚS Y SUBMENÚS BÁSICOS</u>	<u>4</u>
<u>2.1.</u>	<u>EMPLEANDO APPCOMPACTACTIVITY</u>	<u>4</u>
<u>2.2.</u>	<u>RECURSOS IMPLICADOS</u>	<u>9</u>
<u>2.2.1.</u>	<u>AGRUPAMIENTOS</u>	<u>15</u>
<u>2.3.</u>	<u>LANZAR Y PROCESAR EL MENÚ</u>	<u>18</u>
<u>2.4.</u>	<u>CASO PRÁCTICO</u>	<u>18</u>
<u>2.4.1.</u>	<u>LAS IMÁGENES DE LOS MENÚS</u>	<u>20</u>
<u>2.4.2.</u>	<u>RECURSOS XML</u>	<u>21</u>
<u>2.4.3.</u>	<u>EL CÓDIGO DE LA ACTIVITY</u>	<u>22</u>
<u>2.4.4.</u>	<u>ACCESO A UN MENÚ ITEM</u>	<u>23</u>

1. Introducción

- ✓ En esta unidad vamos a ver cómo gestionar menús, submenús y menús contextuales.
- ✓ La siguiente imagen muestra los menús en la **Action Bar** de la aplicación



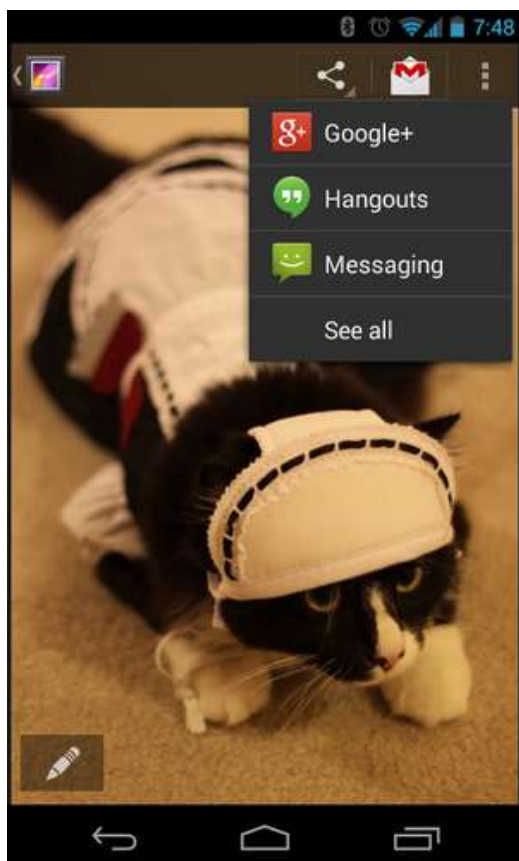
- ✓ 1: Icono de la App y nombre
- ✓ 2: Dos iconos de menús de la aplicación
- ✓ 3: Menú **OverFlow** que equivale al botón Menú de la Botonera
- ✓ Los menús pueden ponerse en la **Action Bar** desde la versión 3.0 de Android.
- ✓ Esta barra pasa a llamarse **AppBar** a partir de la versión Android 5 (API 21).
- ✓ Para versiones anteriores a la 3.0 los menús se veían en la parte inferior de la siguiente imagen:



- ✓ La siguiente imagen muestra 3 botones de menú en la Barra de Acción y el menú Overflow

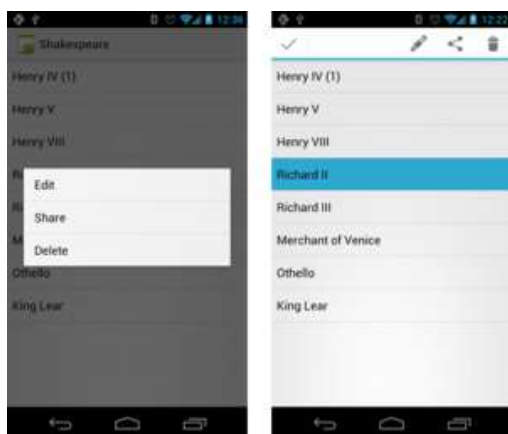


- ✓ En esta imagen se ve como uno de los menús de la actionbar tiene **submenús**:



✓ Finalmente las siguientes imágenes muestran **Menús Contextuales**

- **Flotantes** (izquierda)
- O en la **barra de acciones** (derecha)

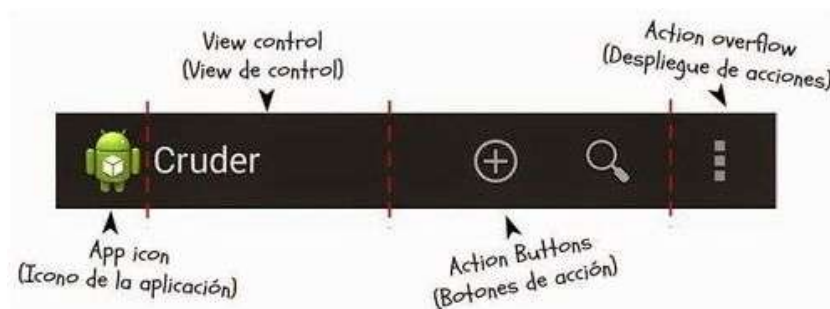


✓ **Referencias:**

- <http://developer.android.com/guide/topics/ui/menus.html>
- <http://developer.android.com/guide/topics/resources/menu-resource.html>
- <http://developer.android.com/guide/topics/ui/actionbar.html>
- <http://developer.android.com/design/patterns/actionbar.html>

2. Menús y submenús básicos

- ✓ Vamos a ver como personalizar la ActionBar (se llama ToolBar y forma [parte de la AppBar](#), a partir de la versión API 21) y que aparezcan opciones de menú:

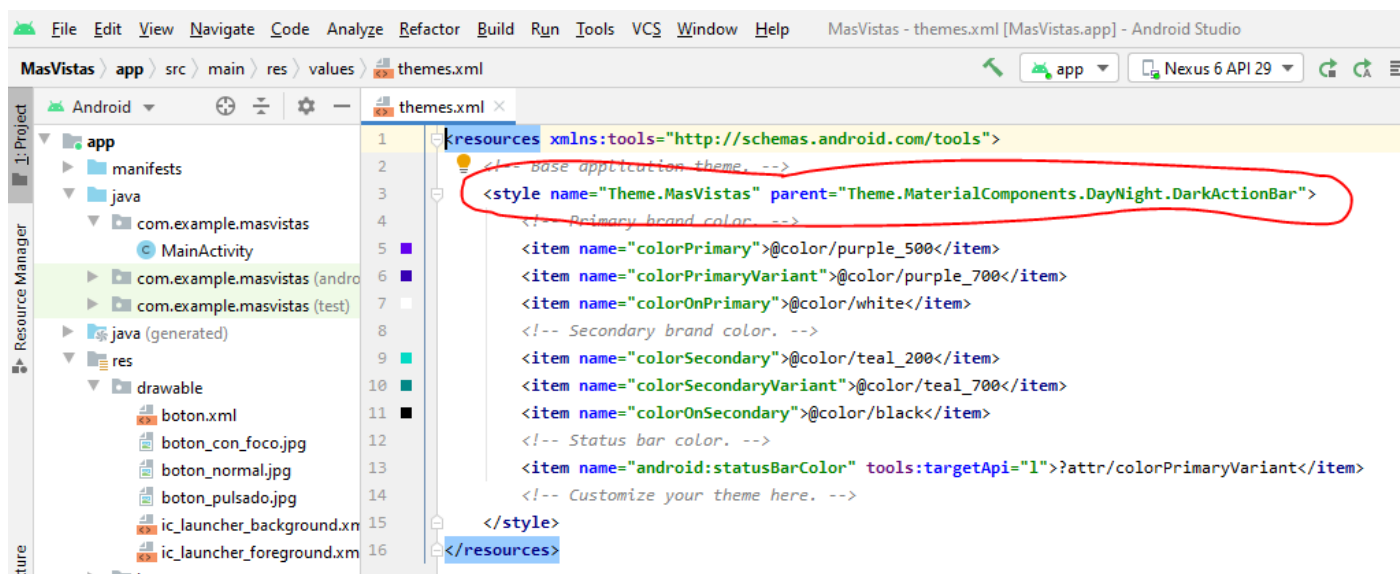


- ✓ Nota:
 - Podemos obtener iconos en el sitio web: <http://www.iconarchive.com/> (intentar utilizar los que tienen licencia free).
- ✓ Vamos a realizar una aplicación con 3 opciones de menú (uno de ellos con submenús y otro con un icono en la Barra de Acción).
- ✓ Si queremos que la ActionBar (ToolBar dentro de la AppBar a partir de la versión API 21):
 - Tenga el mismo aspecto en todas las versiones de Android.
 - Posibilidad de hacer uso de la interface de usuario [Material Design](#) (introducido a partir de la API 21).
- ✓ Tendremos que hacer uso de las bibliotecas de compatibilidad y hacer que nuestra activity derive de AppCompatActivity.
- ✓ Al hacerlo tendremos que cambiar el diseño por uno de tipo AppCompatActivity.

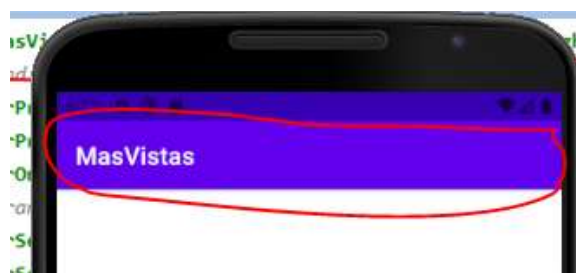
2.1. Empleando AppCompatActivity

- ✓ Se estamos desarrollando un proyecto y vuestras activities derivan de dicha clase para tener compatibilidad con las versiones de Android anteriores, podemos hacer que la AppBar aparezca de dos formas:
 - Empleando un theme (estilo) que incorpore la ActionBar.
 - Empleando un theme **que no incorpore el ActionBar** y añadiendo al Layout de la Activity el View que va representar a **ActionBar (en nuestro caso ToolBar)**.
- ✓ La opción recomendada es la segunda, ya que tenemos un control completo sobre el aspecto e posición de la ActionBar (ToolBar).

Empleando un theme con ActionBar

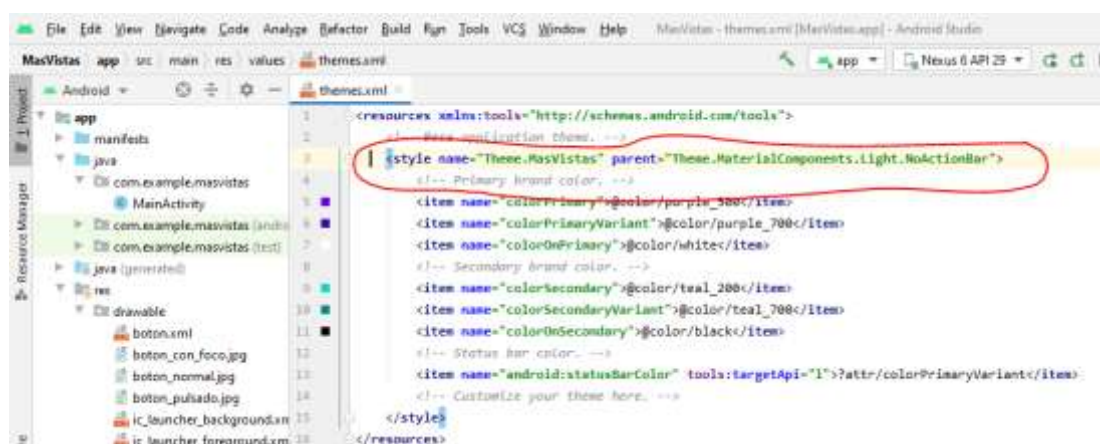


Por defecto, en el theme de la aplicación ya deberíamos tener uno que muestre la ActionBar (ToolBar)



Si ejecutamos podemos ver como aparece la ToolBar.

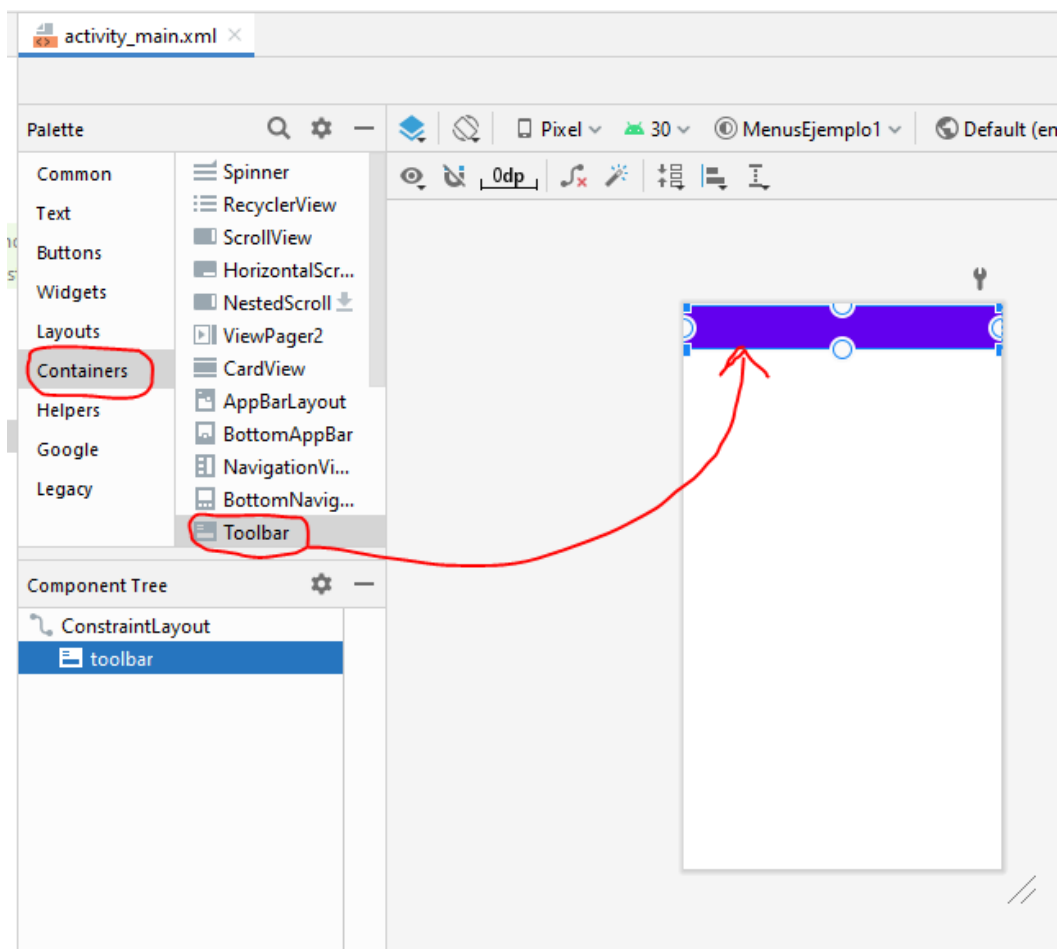
Empleando el view ToolBar



En este caso cambiamos el theme por uno que no muestre un ActionBar.



Si ejecutamos podemos ver como no aparece la Toolbar.



Arrastramos el View Toolbar al Layout de la Activity y la colocamos de acuerdo al Layout que tengamos (en el ejemplo un constraintLayout).

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="?attr/colorPrimary"
        android:minHeight="?attr/actionBarSize"
        android:theme="?attr/actionBarTheme"
        android:elevation="4dp"
        tools:layout_editor_absoluteX="16dp"
        tools:layout_editor_absoluteY="3dp"
        tools:ignore="MissingConstraints" />

</androidx.constraintlayout.widget.ConstraintLayout>

```

Aunque que no aparezca en las propiedades gráficas podemos asignarle una 'elevación' según el Material Design, para que parezca que está por encima del contenido de la Activity. Fijarse que el prefijo que se tiene que poner es **app**: para indicar que emplee la propiedad elevation de la biblioteca de compatibilidad. Si cambiáis por android: estaréis empleando la propiedad que está definida en la API del S.O. Esta propiedad apareció a partir de la API 21, por lo que dará un aviso de que no se empleará en API's inferiores.

```

C MainActivity.java x
1 package com.example.menusejemplo1;
2
3 import androidx.appcompat.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.widget.Toolbar;
6
7 public class MainActivity extends AppCompatActivity {
8
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13
14        Toolbar myToolbar = findViewById(R.id.toolbar_UD06_01_Menus);
15        setSupportActionBar(myToolbar);
16    }
17
18    private void setSupportActionBar(Toolbar myToolbar) {
19    }
20 }

```

A nivel de código tenemos que informar a la activity que emplee dicha ToolBar.

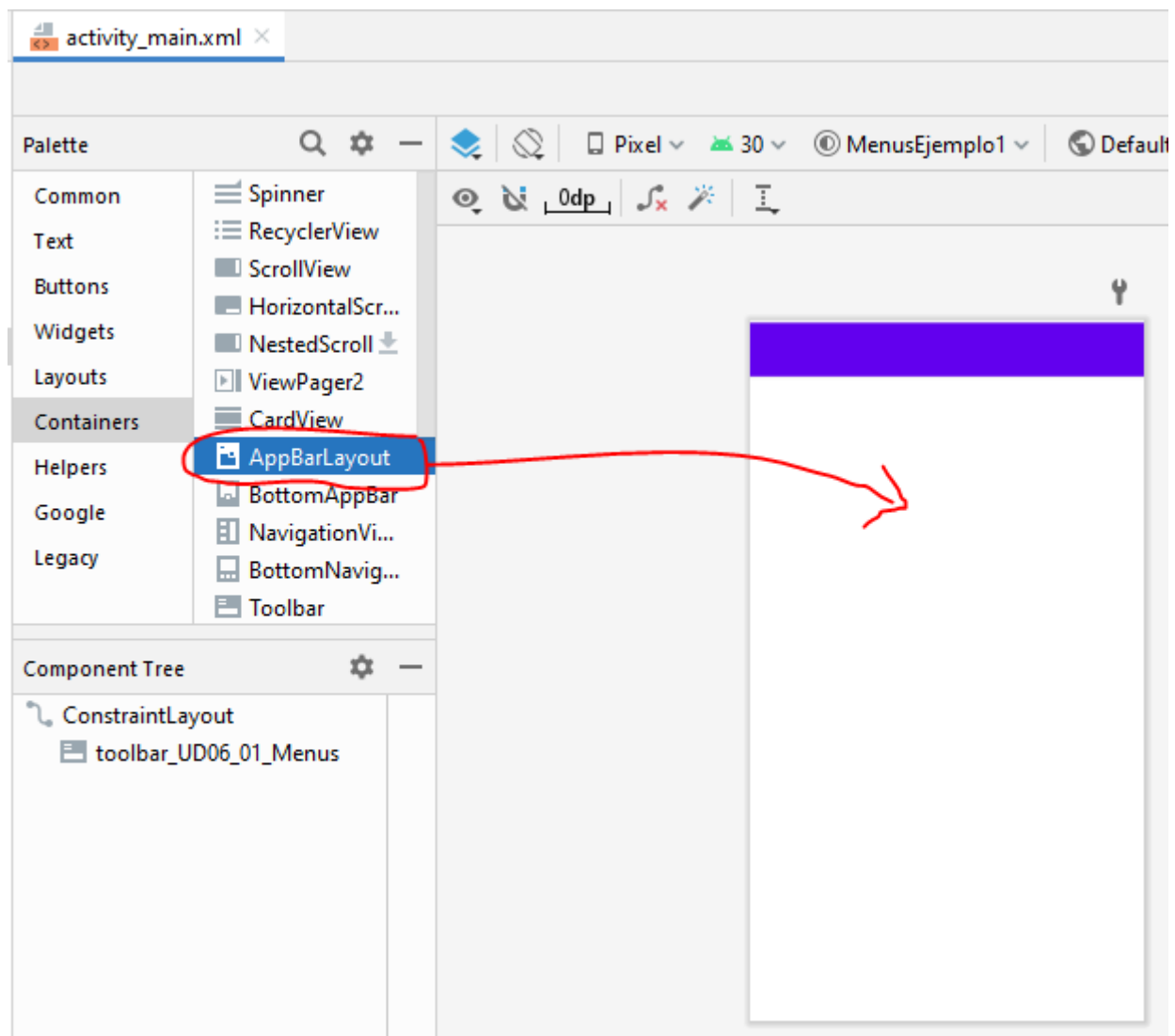
- ✓ El código que hay que añadir en la activity en el método 'onCreate' es:

```
Toolbar myToolbar = findViewById(R.id.toolbar_UD06_01_Menus);  
setSupportActionBar(myToolbar);
```

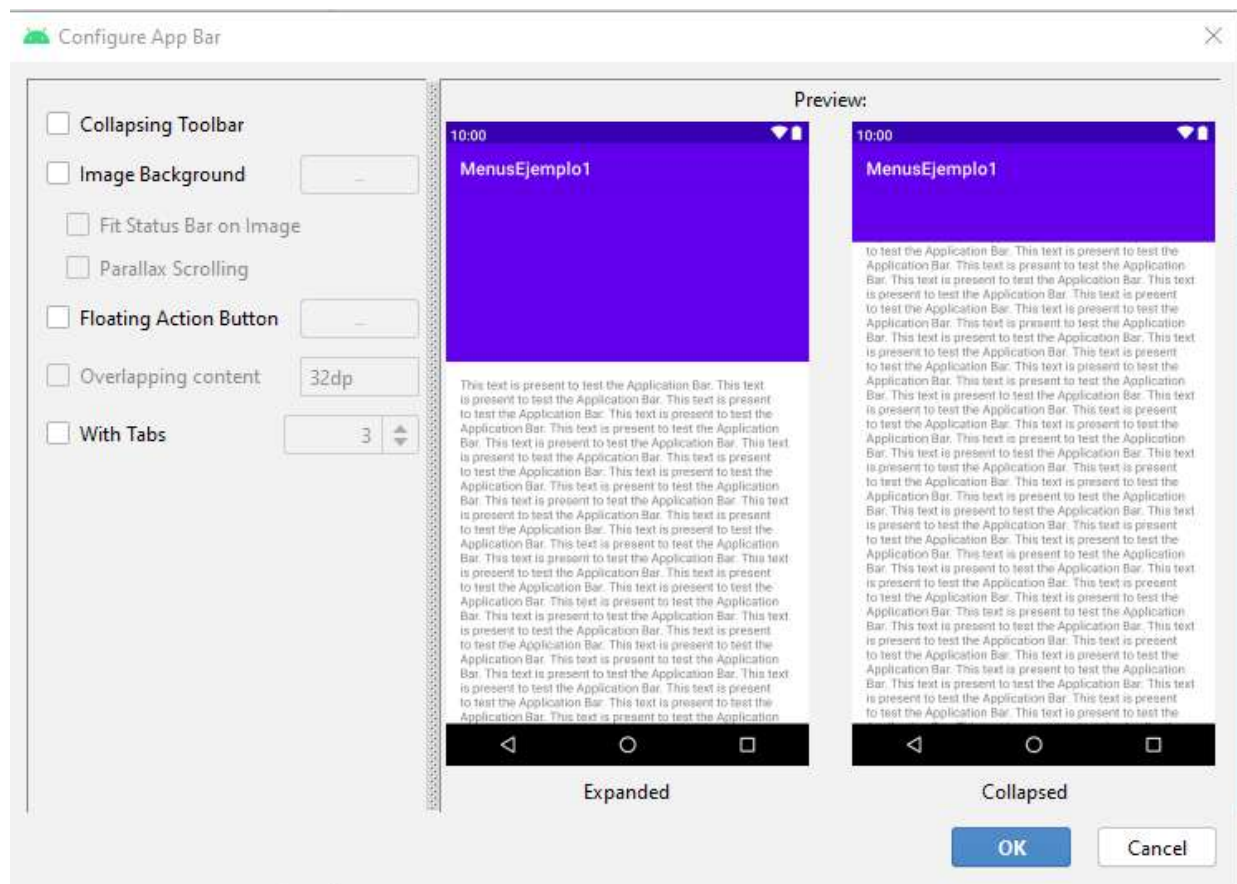
Nota: Indicar que myToolbar representa a Toolbar y podemos aplicarle diferentes métodos como hide() para ocultarla.

- ✓ A nivel del asistente de Android Studio podemos crear una AppBar (que incluye Toolbar) con efectos como hacerse invisible automáticamente cuando hay un scroll, poner imágenes de fondo,...lo único que hay que hacer es arrastrar componentes **appBarLayout** en vez do Toolbar de antes.

Empleando el view AppBar



Arrastramos el componente AppBarLayout

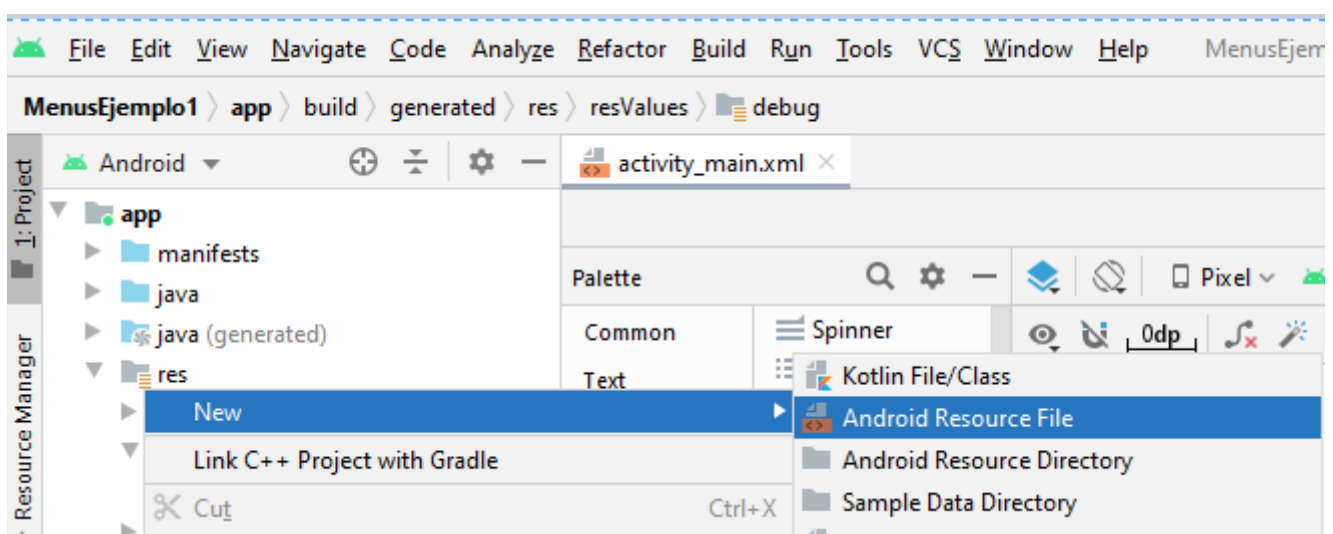


Aparece una pantalla donde indicamos diversos aspectos da AppBar.

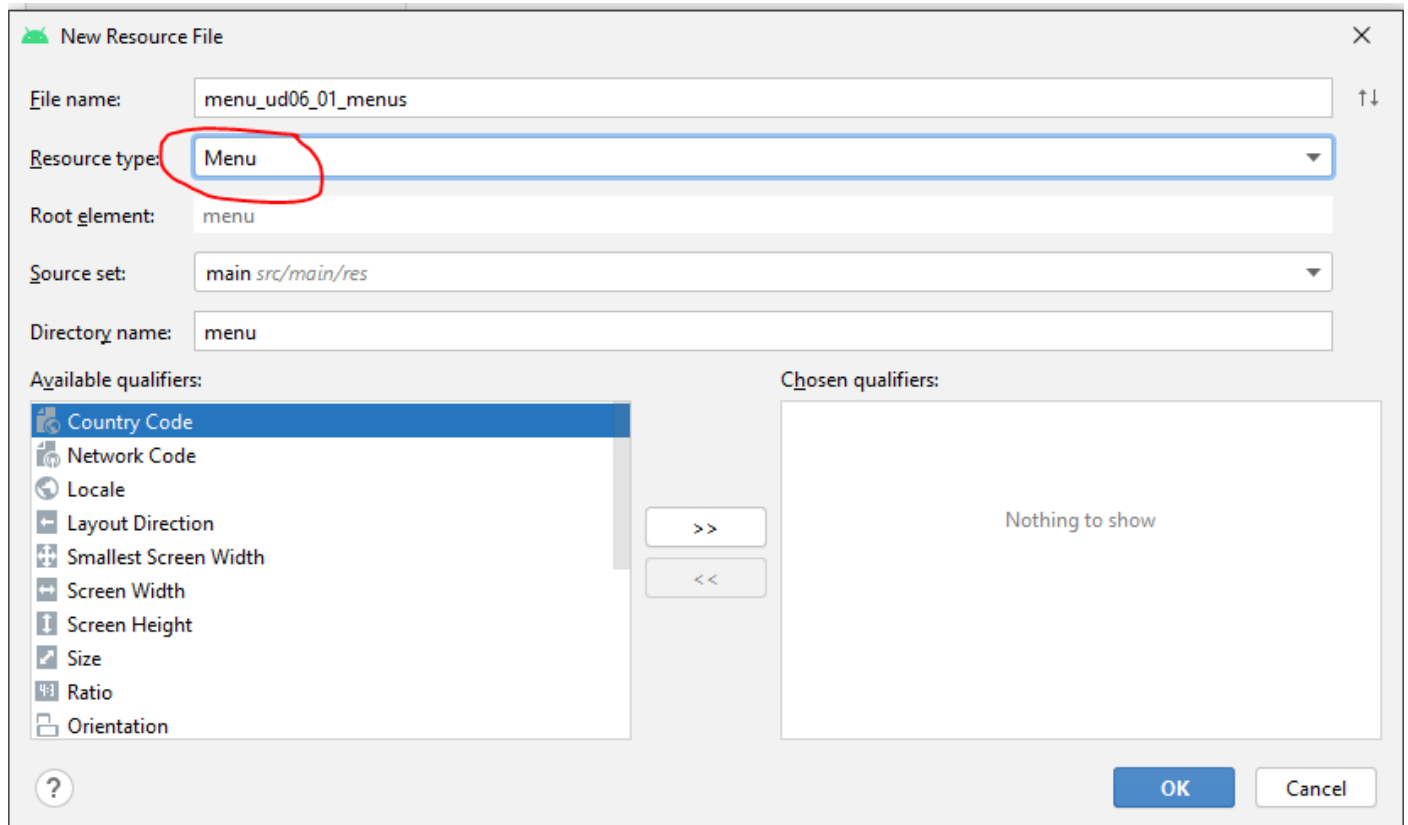
2.2. Recursos Implicados

- ✓ Los menús se definen en un recurso xml en `res/menu/nombre_fichero.xml`

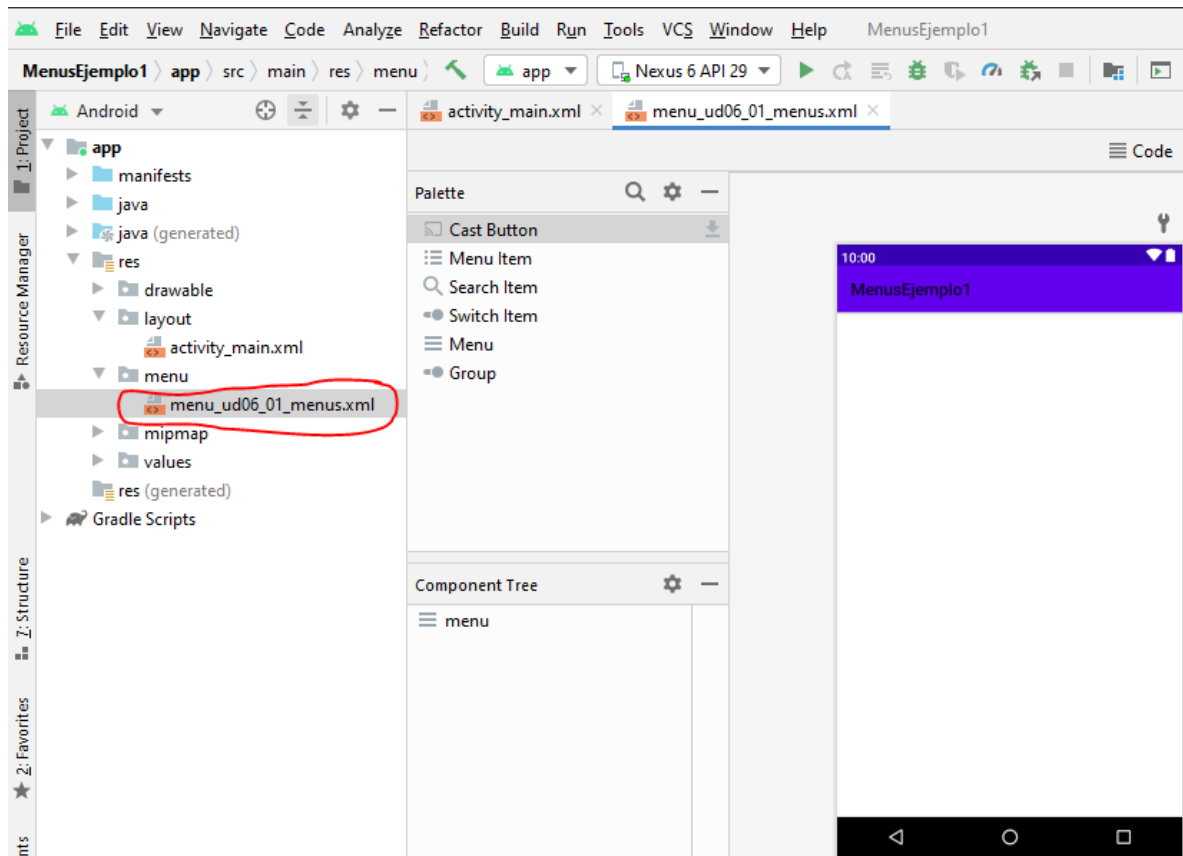
Creando un menú



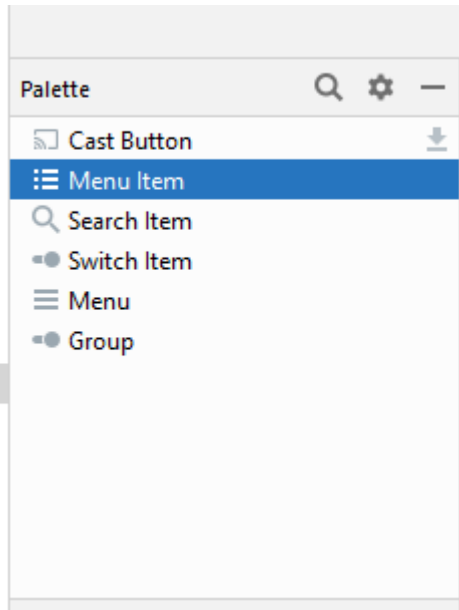
En la rama `/res/` pulsamos el botón derecho del ratón y escogemos crear un nuevo archivo de recursos.



Al crear el recurso a nivel de `/res` podemos elegir el tipo de recurso. En este caso de tipo **menú**.

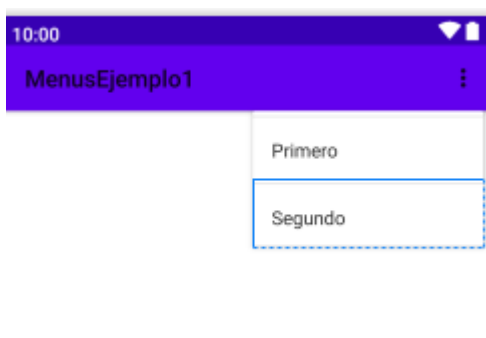


Aparece el archivo creado y también una pantalla de diseño como de layout donde vamos a poder, 'gráficamente' crear un menú.



Podemos 'colocar' los diferentes elementos del menú.

- ✓ Dicho recurso tendrá como elementos:
 - **<menu>**: elemento raíz. Contiene <item> y <group>
 - **<item>**: representa un elemento de menú.
 - Tendría que tener un elemento **<menu>** dentro del <item> para crear un submenú.
- ✓ Arrastrad dos 'menuitem' y cambiad sus propiedades:
 - id: mniPrimero
 - Texto: Primero
 - id:mniSegundo
 - Texto: Segundo



Arrastrad dos de los menuitems, teniendo las propiedades cambiadas.

Creando un menú

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/mniPrimero"
        android:title="Primero"/>

    <item
        android:id="@+id/mniSegundo"
        android:title="Segundo"/>

</menu>
```

El código generado.

IMPORTANTE: Si estamos empleando una AppCompatActivity tendremos que asegurarnos que este definido el espacio de nombres app en la etiqueta 'menu' en el archivo xml, de la forma: `xmlns:app="http://schemas.android.com/apk/res-auto"`

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/mniPrimero"
        android:title="Primero"/>

    <item
        android:id="@+id/mniSegundo"
        android:title="Segundo"/>

</menu>
```

✓ Ahora comentaremos algunos de los atributos que podemos cambiar.

Recordar que podemos hacer gráficamente o directamente en el editor en la vista XML.

- Atributo **android:icon**: Icono asociado a esta opción de menú. Recordar que ya vimos anteriormente anteriormente como hacer uso do Image Asset Store. Ahora deberemos escoger la categoría 'Action bar and Tab Icons'.
- Atributo **android:showAsAction** (si la activity **deriva da clase Activity**):
- Atributo **app:showAsAction** (si la activity **deriva da clase AppCompatActivity**)


IMPORTANTE: Si tenemos importada la librería de compatibilidad AppCompatActivity en el build.gradle nos va a obligar a emplear esta opción (app:) y por lo tanto tendremos que hacer que la activity derive de AppCompatActivity.

<input checked="" type="checkbox"/> showAsAction	<input type="checkbox"/>	
always	<input type="checkbox"/> false	
never	<input type="checkbox"/> false	
ifRoom	<input type="checkbox"/> false	
collapseActionView	<input type="checkbox"/> false	
withText	<input type="checkbox"/> false	

✓ Puede tener los siguientes valores:

- **never**: nunca muestra el item del menú en la barra de acción.
- **ifRoom**: se hay espacio en la barra de acción lo muestra.
- **always**: muestra siempre en la barra de acción.
- **withText**: si tenemos un icono, por defecto no muestra el texto. Se queremos que muestre los dos marcaremos este valor.
- **collapseActionView**: cuando la acción de un elemento de menú (declarada coma **android:actionLayout** o **android:actionViewClass**) es plegable. Disponible a partir de la versión API 14. Ejemplo de uso [en este enlace](#).

- Atributo **android:orderInCategory="Nº Entero"**: permite ordenar las acciones según el número indicado. Las acciones con un número más pequeño se sitúan más arriba. Para establecer el valor de la propiedad podemos hacerlo:
 - Utilizando 'constantes' que ya tienen un valor asignado.



```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/mniPrimero"
        android:orderInCategory="@android:integer/config_shortAnimTime"
        android:title="Primero" />
    <item
        android:id="@+id/mniSegundo"
        android:orderInCategory="@android:integer/config_mediumAnimTime"
        android:title="Segundo" />
    <item
        android:id="@+id/mniTercero"
        android:orderInCategory="@android:integer/config_shortAnimTime"
        android:title="Tercero" />
</menu>
```

- Utilizando valores numéricos enteros.



```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/mniPrimero"
        android:orderInCategory="10"
        android:title="Primero" />
    <item
        android:id="@+id/mniSegundo"
        android:orderInCategory="20"
        android:title="Segundo" />
    <item
        android:id="@+id/mniTercero"
        android:orderInCategory="10"
        android:title="Tercero" />
</menu>
```

Este será el resultado:

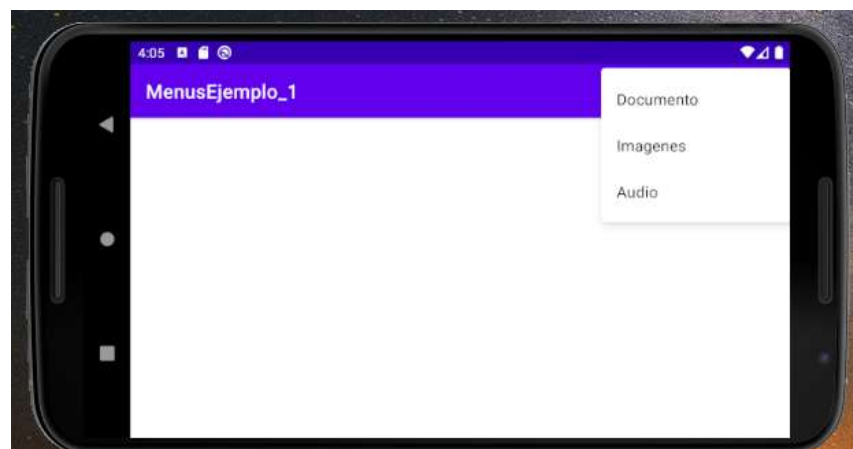
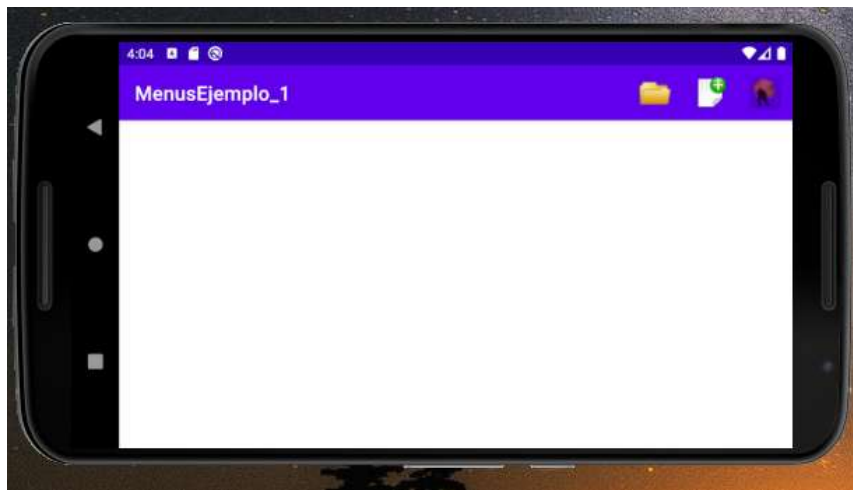


Observar que si todos o alguno tienen el mismo valor o no se pone valor alguno entonces aparece en la orden en que son creados, de izquierda a derecha y de arriba-abajo.

- ✓ Los elementos de menú que aparecen en la barra, ya no aparece en el menú al pulsar el botón Menú u "OverFlow", aunque estén en un group, como se puede apreciar en las imágenes de abajo.
- ✓ Los valores se pueden combinar con el carácter '|'.

✓ EJEMPLO DE MENÚ EN LA BARRA DE ACCIÓN CON SUBMENUS

- ✓ La siguiente imagen, se obtiene con el siguiente recurso xml.



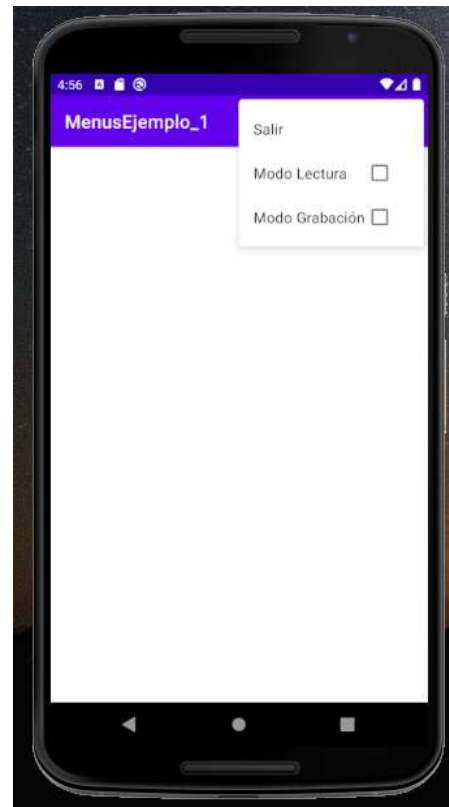
- ✓ Disponéis de las imágenes en el archivo adjunto 'PDM_icons_menu_ejemplo.zip'.
- ✓ Líneas 6, 27, 34 (**android:icon**): Indican el recurso *drawable* que deben mostrar. Estas imágenes deben estar en la carpeta */res/drawableXXXXX* correspondientes.
- ✓ Líneas 7, 28, 35 (**app:showAsAction**): Indican que si hay espacio en la barra de acción que muestren esos ítems que tienen ese atributo.
- ✓ Líneas 9, 30, 37 (**android:titleCondensed**): El atributo **android:titleCondensed** se usará cuando el valor de **android:title** sea demasiado largo.
- ✓ Líneas 10-23 **<menu>**: Para crear submenús dentro de un menú, volvemos a crear la entrada **<menu>** dentro de **<item>**.

2.2.1. Agrupamientos

- ✓ Otro elemento que puede ir dentro del elemento **<menu>**:
 - **<group>**: un grupo es un conjunto de elementos que tienen ciertas características.
 - Mostrar u ocultar todos los elementos: **setGroupVisible()**
 - Habilitar o deshabilitar todos los elementos: **setGroupEnabled()**
 - Hacer que todos los ítems del group puedan ser presentados en forma de checkbox: **setGroupCheckable()**

- ✚ Podemos hacer que un elemento sea chequeable usando el atributo **android:checkable** en el elemento <item> o el
 - ✚ group entero con 'android:checkableBehavior' en el elemento <group>.
 - ✚ El atributo **android:checkableBehavior** puede tener las siguientes opciones:
 -1. **single**: solo un elemento del grupo puede ser chequeado (radiobuttons).
 -2. **all**: todos los elementos pueden ser chequeados (checkboxs).
 -3. **none**: ningún elemento es seleccionable.
 - ✚ Sino no ponemos nada, los elementos del <group> aparecen como los demás.
- ✓ La siguiente imagen, tiene asociado el siguiente XML.
 - ✓ Observar que en la barra de acción no entran todos los menús.
 - ✓ Observar que hay dos menús (Modo Lectura y Modo Grabación) que pueden ser marcados los dos.

Menultems en el AppBar



No caben todos los elementos de menú.

Al pulsar en los puntos suspensivos del AppBar aparecen los demás elementos.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/itemAbrir"
        android:icon="@drawable/open32"
        app:showAsAction="ifRoom"
```



```

        android:title="Abrir"
        android:titleCondensed="Abrir">
        <menu>
            <item
                android:id="@+id/itemDoc"
                android:title="Documento"
                android:titleCondensed="Doc."/>
            <item
                android:id="@+id/itemIma"
                android:title="Imagenes"
                android:titleCondensed="Imag."/>
            <item
                android:id="@+id/itemAudio"
                android:title="Audio"
                android:titleCondensed="Audio"/>
        </menu>
    </item>
    <item
        android:id="@+id/itemNuevo"
        android:icon="@drawable/new32"
        app:showAsAction="ifRoom"
        android:title="Nuevo documento"
        android:titleCondensed="N.Doc.">
    </item>
    <item
        android:id="@+id/itemSalir"
        android:icon="@drawable/exit32"
        app:showAsAction="ifRoom"
        android:title="Salir"
        android:titleCondensed="Salir">
    </item>

    <group
        android:id="@+id/mgrpModos"
        android:checkableBehavior="all" >
        <item
            android:id="@+id/ModoLect"
            app:showAsAction="ifRoom"
            android:title="Modo Lectura"
            android:titleCondensed="M.Lect.">
        </item>
        <item
            android:id="@+id/ModoRecord"
            app:showAsAction="ifRoom"
            android:title="Modo Grabación"
            android:titleCondensed="M.Grab">
        </item>
    </group>

</menu>

```

- ✓ Líneas 35, 45, 51: Observar como estos ítems no se muestran en la barra de acción porque no tienen espacio.
- ✓ Líneas 40-55: Observar como hay un agrupamiento de ítems y en la línea 42 se indica que todos ellos son "chequeables".

2.3. Lanzar y procesar el menú

- ✓ Para lanzar el menú debemos cambiar el menú que tienen asignada una determinada Activity y para eso tenemos que sobrescribir el [método onCreateOptionsMenu\(\)](#).

- ✓ Lo que hacemos es "inflar" el xml asociado al menú (Línea 4) del mismo hecho que se hace cuando se

```
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(menu_ud06_01_menus, menu);
    return true;
}
```

- ✓ Cuando se pulse un ítem de un menú se lanza el evento onOptionsItemSelected y para procesar ese ítem sobrescribir el [método onOptionsItemSelected\(\)](#).

- ✓ Este método devuelve un boolean:

- **true**: se procesamos un elemento do menú (return true).
- **false**: si queremos que el evento siga 'propagándose' por el resto de la jerarquía de views del Activity.

- ✓ Si tenemos los elementos de menú como de tipo 'seleccionable' dentro de un grupo, debemos ser nosotros los que marquemos llamando al método setChecked (boolean valor) y para saber se está chequeado o método getChecked().

```
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.

    int id = item.getItemId();
    if (id == R.id.itemAbrir) {
        return true;
    }
    return super.onOptionsItemSelected(item);
}
```

2.4. Caso práctico

- ✓ Crearemos un nuevo paquete de nombre: **Menus** como un subpaquete de tu paquete principal.
- ✓ Dentro del paquete **Menus** crear una nueva 'Empty Activity' de nombre: **UD06_01_Menus** de tipo Launcher y sin compatibilidad. Modifica el archivo **AndroidManifest.xml** y añade una label a la activity.

Menús básicos



Observar como tenemos un icono en la barra de acción (la cara) es el menú "overflow"

Si pulsamos en la cara se realiza una acción, en este caso se cambia el texto del TextView y se lanza un Toast.



Si pulsamos en el menú "overflow" obtenemos otros 2 menús ...

Se pulsamos en cualquiera de los dos casos anteriores en el menú "Opción2" se lanza un Toast y se cambia el texto del TextView



Si pulsamos en la "Opción 3 - Submenú" obtenemos los submenús asociados a ese ítem.

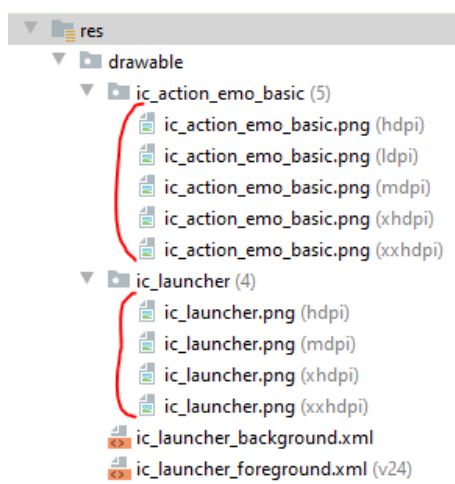
En este caso se cerró la app y se mostró un Toast.

Podemos:

- Cambiar el texto del textView.
- Salir de la aplicación.

2.4.1. Las imágenes de los menús

- ✓ La imagen de la cara está disponible en el archivo adjunto 'Res.zip', una vez descomprimida se puede copiar tal cual al proyecto en la carpeta 'res'.



- ✓ Estas imágenes y otras pueden obtenerse de:
 - <http://www.androidicons.com/>
 - <http://www.glyfx.com/content/page/android-common-ii.html>

2.4.2. Recursos XML

- ✓ El xml del Layout: **activity_ud06_01__menus**

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".UD06_01_Menus">

    <TextView
        android:id="@+id/tvTexto_UD06_01_Menus"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="TextView"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

- ✓ Creamos un XML con el texto de las opciones de menú en /res/values/ de nombre **strings_mnu_ud06_01_menus**:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="str_mniItem1_UD06_01_Menus">Opción 1</string>
    <string name="str_mniItem2_UD06_01_Menus">Opción 2</string>
    <string name="str_mnuItem3_UD06_01_Menus">Opción 3- Submenús</string>
    <string name="str_mniItem3_1_UD06_01_Menus">Opción 3- Submenú 1 </string>
    <string name="str_mniItem3_2_UD06_01_Menus">Opción 3- Submenú 2 - Salir</string>
</resources>
```

- ✓ El xml del 'menú' (modificamos el recurso XML creado por defecto en /res/menu/...): **mnu_ud06_01_menus.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/mniItem1_04_01_Menus"
        android:icon="@drawable/ic_action_emo_basic"
        android:orderInCategory="100"
        app:showAsAction="always"
        android:title="@string/str_mniItem1_UD06_01_Menus" />
    <item
        android:id="@+id/mniItem2_04_01_Menus"
        android:orderInCategory="100"
```

```

        app:showAsAction="never"
        android:title="@string/str_mniItem2_UD06_01_Menus"/>
    <item
        android:id="@+id/mnuItem_3_04_01_Menus"
        android:orderInCategory="100"
        app:showAsAction="never"
        android:title="@string/str_mnuItem3_UD06_01_Menus">
        <menu>
            <item
                android:id="@+id/mniItem3_1_06_01_Menus"
                android:title="@string/str_mniItem3_1_UD06_01_Menus"/>
            <item
                android:id="@+id/mniItem3_2_06_01_Menus"
                android:title="@string/str_mniItem3_2_UD06_01_Menus"/>
            </menu>
        </item>
    </menu>

```

- ✓ **Líneas 8, 13, 18:** Observar como todos los ítems tienen el mismo valor para ser ordenados.
 - Probad a cambiar el tercer ítem a un valor de 99.
- ✓ **Líneas 9, 14, 19:** Observar cómo el primer ítem indica *always* y los demás *never* a la hora de mostrar el menú en la barra de acción.

2.4.3. El código de la Activity

```
package com.example.ud06_01_menus;
```

```

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView;
import android.widget.Toast;

```

```
public class UD06_01_Menus extends AppCompatActivity {
```

```

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_u_d06_01__menus);
    }

```

```

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.mnu_ud06_01_menus, menu);
        return true;
    }

```

```

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
    }

```

```

        TextView tv = (TextView) findViewById(R.id.tvTexto_UD06_01_Menus);
        Toast.makeText(getApplicationContext(), "Pulsado elemento: " +
item.getTitle().toString(), Toast.LENGTH_SHORT).show();

        switch (item.getItemId()) {
            case R.id.mniItem1_06_01_Menus:
                tv.setText("Quedaré con esta cara de contenta cuando termine la tarea");
                return true;

            case R.id.mniItem2_06_01_Menus:
                tv.setText("La opción 2 no tiene nada asignado");
                return true;

            case R.id.mniItem3_1_06_01_Menus:
                tv.setText("Has seleccionado el submenú 1 de la opción 3");
                return true;

            case R.id.mniItem3_2_06_01_Menus:
                finish();
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }
}

```

- ✓ **Línea 37:** Recogemos el título del ítem pulsado.
 - ✓ **Líneas 42, 46, 52, 56:** devolver true en el caso de procesar cada uno de los ítems.
 - ✓ **Línea 58:** Si no se proceso el ítem llamar al padre, que va a devolver false por defecto.
- ✓ Si en la definición de cada ítem se hubiera usado el atributo **android:onClick** entonces tendríamos que crear en java los métodos correspondientes del mismo hecho que se hace cuando se usa ese atributo en un Layout.

2.4.4. Acceso a un menú item

- ✓ Si queremos acceder a un menú item dinámicamente fuera del evento `onOptionsItemSelected`, podemos:
 - * Guardar una referencia al objeto menu inflado en el método `onCreateOptionsMenu`, y después llamar al método `findItem(int resId)` para obtener la referencia al elemento del menú.
 - Si tenemos acceso al toolbar, podemos hacer un `findViewById` del toolbar y llamar al método `getMenu()`.

Suponemos que tenemos una activity con un elemento de menú y queremos habilitarlo o inhabilitarlo en función del estado de un `ToggleButton`:

```

public class MenuActivity extends AppCompatActivity {

    private Menu menu;

    @Override

```

```
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()) {
        case R.id.mniPrimero:
            Toast.makeText(this, "PREMIDO ELEM 1", Toast.LENGTH_SHORT).show();
            break;
    }
    return true;
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {

    getMenuInflater().inflate(R.menu.menu_activity, menu);
    this.menu = menu;

    return true;
}

private void xestionarEventosBotons() {
    findViewById(R.id.tgbtnHabilitarMenuItem).setOnClickListener(
        new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                ToggleButton tgbutton = (ToggleButton) view;
                MenuItem menuItem = menu.findItem(R.id.mniPrimero);
                menuItem.setEnabled(tgbutton.isChecked());
            }
        }
    );
}

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_menu);

    Toolbar toolbar = findViewById(R.id.toolbar);
    setSupportActionBar(toolbar);

    xestionarEventosBotons();
}
}
```