1 Menús

1.1 Sumario

1.2 Introducción

- Nesta unidade imos ver como xestionar menús, submenús e menú contextuais.

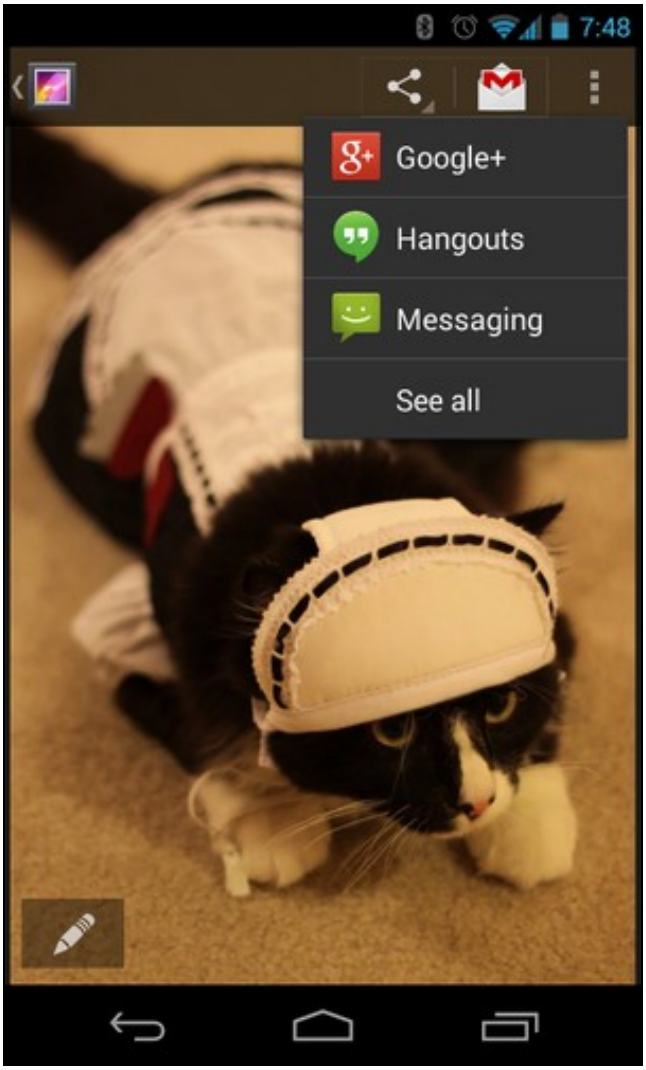
- A seguinte imaxe amosa os menús na **Action Bar** da aplicación

- **1**: Icona da App e nome
- **2**: Dúas iconas de menús da aplicación
- **3**: Menú **OverFlow** que equivale ao botón Menú da Botonera.
- Os menús poden poñerse na **Action Bar** dende a versión 3.0 de Android.

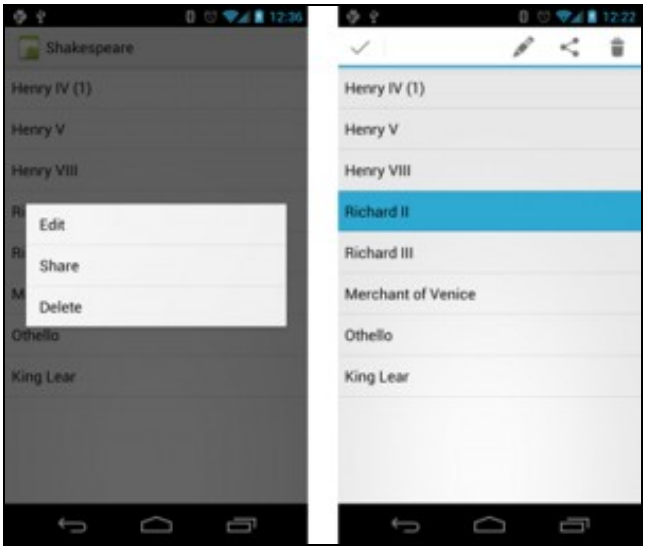
- Para versións anteriores á 3.0 os menús víanse como na parte inferior da seguinte imaxe:

- A seguinte imaxe amosa 3 botóns de menú na Barra de Acción e o menú Overflow

- Nesta imaxe vese como un dos menús da action bar ten **submenús**:



- Finalmente as seguintes imaxes amosan **Menús Contextuais**
  - **Flotantes** (esquerda)
  - ou na **barra de accións** (dereita)



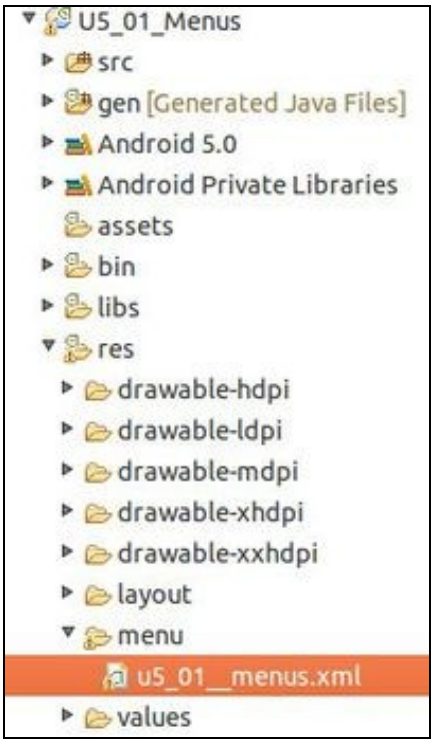
- **Referencias**:
    - http://developer.android.com/guide/topics/ui/menus.html
    - http://developer.android.com/guide/topics/resources/menu-resource.html
    - http://developer.android.com/guide/topics/ui/actionbar.html
    - http://developer.android.com/design/patterns/actionbar.html

1.3 Menus e submenús básicos

- Imos realizar unha aplicación con 3 menús (un deles con submenús e outro cunha icona na Barra de Acción).
- Antes imos ver distintas casuísticas:

1.3.1 Recursos Implicados

- Os menús defínense nun recurso xml en **res/menu/nome_ficheiro.xml**



- Dito recurso terá coma elementos:
    - **<menu>**: elemento raíz. Contén <item> e <group>
    - **<item>**: representa un elemento de menú.
        - ◊ Tería que ter un elemento **<menu>** dentro de <item> para crear un submenú.

- O xml seguinte é o que se crea por defecto cando se crea un novo proxecto:

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context="com.example.u5_01_menus.U5_01_Menus" >

    <item
        android:id="@+id/action_settings"
        android:orderInCategory="100"
        android:showAsAction="never"
        android:title="@string/action_settings"/>

</menu>
```

- **Liña 7**: o atributo **android:orderInCategory="Nº Enteiro"**. Se hai varios ítems indica cal aparecerá primeiro, segundo, etc. Se todos teñen o mesmo valor entón aparecen na orde na que son creados.

- **Liña 8**: o atributo **android:showAsAction** pode ter os seguintes valores:
    - **never**: nunca amosa o item de menú na barra de acción.
    - **ifRoom**: se hai espazo na barra de acción o amosa.
    - **always**: amosa sempre na barra de acción.
    - **withText**: se temos unha icona, por defecto non amosa o texto. Se queremos que amose os dúos poñeremos este valor.
    - **collapseActionView**: cando a acción dun elemento de menú (declarada coma **android:actionLayout** ou **android:actionViewClass**) é plegable. Dispoñible a'partires da versión API 14.
    - Os elementos de menú que aparecen na barra, xa non aparecen no menú o premer o botón Menú ou "OverFlow", aínda que estean nun group, como se pode apreciar nas imaxes de abaixo.
    - Os valores pódense combinar có caracter ?|?.

- **Liña 9**: o atributo **android:title** indica o nome do menú/submenú. Neste caso o seu valor defínese nun recurso xml de strings:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">U5_01_Menus</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>

</resources>
```

- O atributo **android:onClick** funciona igual que nas vistas do recurso xml do Layout.

- **EXEMPLO DE MENÚ NA BARRA DE ACCIÓN CON SUBMENUS**
- A seguinte imaxe, obténse co seguinte recurso xml.


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

    <item
        android:id="@+id/itemAbrir"
        android:icon="@drawable/open32"
        android: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="Imaxes"
                android:titleCondensed="Imax."/>
            <item
                android:id="@+id/itemAudio"
                android:title="Audio"
                android:titleCondensed="Audio"/>
        </menu>
    </item>
    <item
        android:id="@+id/itemNovo"
        android:icon="@drawable/close32"
        android:showAsAction="ifRoom"
        android:title="Novo documento"
        android:titleCondensed="N.Doc.">
    </item>
    <item
        android:id="@+id/itemSair"
        android:icon="@drawable/exit32"
        android:showAsAction="ifRoom"
        android:title="Saír"
        android:titleCondensed="Saír">
    </item>

</menu>
```

- **Liñas 6,27,34**: Indican o recurso *drawable* que deben amosar. Estas imaxes deben estar na carpeta /**res**/**drawableXXXXX** correspondentes.
- **Liñas 7,28,35**: Indican que se hai espazo na barra de acción que amosen eses ítems que teñen ese atributo.
- **Liñas 9,30,37**: O atributo **android:titleCondensed** usarase cando o valor de **android:title** sexa demasiado longo.
- **Liñas 10-23**: Para crear submenús dentro dun menú, volvemos a crear a entrada <menu> dentro de <item>

1.3.1.1 Agrupamentos

- Outro elemento que pode ir dentro do elemento **<menu>**:
    - ♦ **<group>**: un grupo é un conxunto de elementos que teñen certas características.
        - ◊ Amosar ou ocultar todos os elementos: **setGroupVisible()**
        - ◊ Habilitar ou deshabilitar todos os elementos: **setGroupEnabled()**
        - ◊ Facer que todos os ítems do group poidan ser presentados en forma de checkbox: **setGroupCheckable()**
            - · Podemos facer que un elemento sexa chequeable usando o atributo **android:checkable** no elemento <item> ou o
            - · group enteiro con '**android:checkableBehavior** no elemento <group>.
            - · O atributo **android:checkableBehavior** pode ter as seguintes opcións:
                - **single**: só un elemento do grupo pode ser checkeado (radiobuttons).
                - **all**: todos os elementos poden ser checkeados (checkboxs).
                - **none**: ningún elemento é seleccionable.
            - · Se non poñemos nada, os elementos do <group> aparecen como os demais.

- A seguinte imaxe, ten asociado o seguinte XML.
- Observar que na barra de acción non collen todos os menús.
- Observar que hai dous menús (Modo Lectura e Modo Gravación) que poden ser marcados os dous.


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

    <item
        android:id="@+id/itemAbrir"
        android:icon="@drawable/open32"
        android:showAsAction="always"
        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="Imaxes"
                android:titleCondensed="Imax."/>
            <item
                android:id="@+id/itemAudio"
                android:title="Audio"
                android:titleCondensed="Audio"/>
        </menu>
    </item>
    <item
        android:id="@+id/itemNovo"
        android:icon="@drawable/close32"
        android:showAsAction="ifRoom"
        android:title="Novo documento"
        android:titleCondensed="N.Doc.">
    </item>
    <item
        android:id="@+id/itemSair"
        android:icon="@drawable/exit32"
        android:showAsAction="ifRoom"
```

```
        android:title="Saír"
        android:titleCondensed="Saír">
    </item>

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

</menu>
```

- **Liñas 35,45,51**: Observar como estes ítems non se amosan na barra de acción porque non teñen espazo.
- **Liñas 40-55**: Observar como hai un agrupamento de ítems e na **liña 42** indícase que todos eles son "chequeables".

1.3.2 Lanzar e procesar o menú

- Para lanzar o menú debemos cambiar o menú que ten asinada unha determinada Activity e para iso temos que sobrescribir o método **onCreateOptionsMenu()**.
- Este método xa aparece na Activity cando a creamos con Eclipse.
- O que facemos en "inflar" o xml asociado ao menú (Liña 4) do mesmo xeito que se fai cando se

```
@Override
        public boolean onCreateOptionsMenu(Menu menu) {
                // Inflate the menu; this adds items to the action bar if it is present.
                getMenuInflater().inflate(R.menu.u5_01__menus, menu);
                return true;
        }
```

- Cando se preme un ítem dun menú lánzase o evento onOptionsItemSelected e para procesar ese ítem sobrescribir o método **onOptionsItemSelected(MenuItem item)**.
- Este método tamén se crea por defecto ao crear o proxecto.

```
@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.

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

- Este método devolve un boolean:
    - ♦ true: se procesamos un elemento do menú (return true).
    - ♦ false: se non o procesamos (chamar ao método pai).

1.3.3 Caso p?actico

- Crear o proxecto: **U5_01_Menus**

- Menús básicos
- 

Observar como temos unha icona na barra de acción (a cara) e o menú "overflow"
- 

Se prememos na cara realízase unha acción, neste caso cambiouse o texto do TextView e lanzaouse un Toast.

- 

Se prememos no menú "overflow" obtemos outros 2 menús ...

- 

Os mesmos que se premmos no botón menú da Botonera dos dispositivos que a teñan.

- 

Se prememos en calquera dos dous casos anteriores no menú "Opción2" lánzase un Toast e cambiouse o texto do TextView.

- 

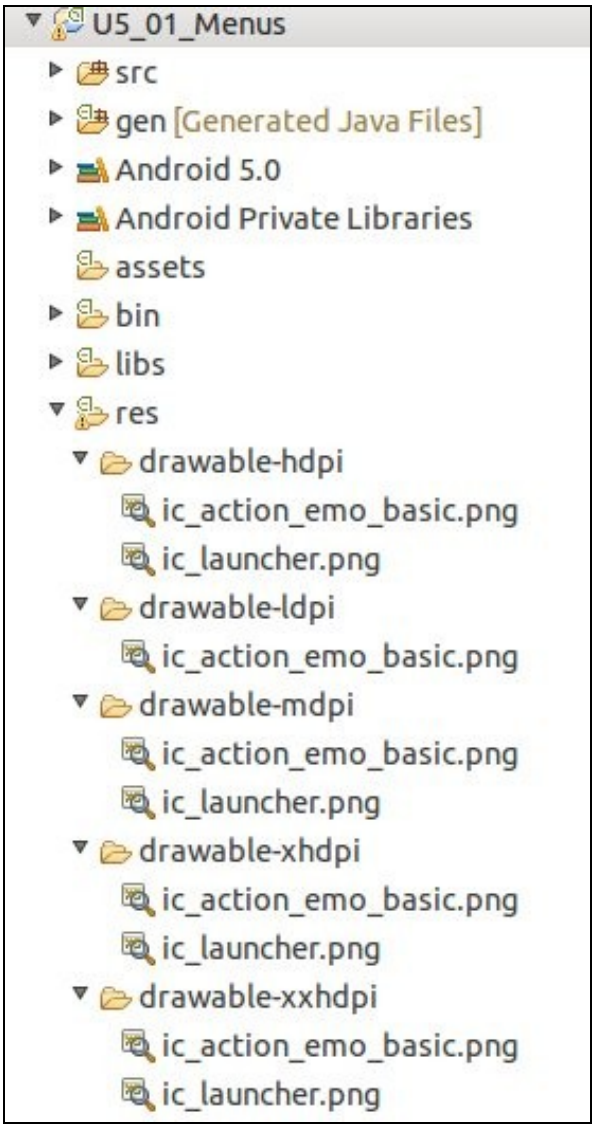
Se prememos na "Opcion 3 - Submenús" obtemos os submenús asociados a ese ítem.
Podemos:
Lanzar a calculadora ou
saír da aplicación.

- 

Neste caso lanzouse a calculadora e un Toast.

1.3.3.1 As imaxes dos menús

- Neste caso colocouse en cada recurso **res**/**drawable** a imaxe da cara:



- Esta imaxe pode obterse do seguinte zip: media:Res.zip e unha vez descomprimido pódese copiar tal cal ao proxecto a "res"
- Estas imaxes e outras poden obterse de:
    - http://www.androidicons.com/
    - http://www.glyfx.com/content/page/android-common-ii.html

1.3.3.2 Recursos XML

- O xml do **Layout**:

- É o creado por defecto salvo que lle engadimos un id (Liñas 11).

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin">

    <TextView
        android:id="@+id/tv"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

</RelativeLayout>
```

- O xml do **menú** (modificamos o recurso XML creado por defecto en /res/menu/...)

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context="com.example.u5_01_menus.U5_01_Menus" >

    <item
        android:id="@+id/item1"
        android:icon="@drawable/ic_action_emo_basic"
        android:orderInCategory="100"
        android:showAsAction="always"
        android:title="@string/item1"/>
    <item
        android:id="@+id/item2"
        android:orderInCategory="100"
        android:showAsAction="never"
        android:title="@string/item2"/>
    <item
        android:id="@+id/item3"
        android:orderInCategory="100"
        android:showAsAction="never"
        android:title="@string/item3">
        <menu>
            <item
                android:id="@+id/SubItem3_1"
                android:title="@string/item3_1"/>
            <item
                android:id="@+id/SubItem3_2"
                android:title="@string/item3_2"/>
        </menu>
    </item>
</menu>
```

- **Liñas 8,13,18**: Observar como todos os ítems teñen o mesmo valor para a ser ordenados.
    - ♦ Que o participante no curso probe a cambiar o terceiro ítem a un valor de 99.
- **Liñas 9,14,19**: Observar como o primeiro ítem indica always e os demais never á hora de amosar o menú na barra de acción.

- Recurso xml de **strings**
- Aínda que podíamos poñer os Títulos dos menús directamente no ficheiro anterior, decidimos, nesta ocasión, realizalo da maneira recomendada: definir os seus valores nun recurso xml.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">U5_01_Menus</string>
    <string name="hello_world">Hello world!</string>
    <string name="item1">Opción 1</string>
    <string name="item2">Opción 2</string>
    <string name="item3">Opción 3- Submenús</string>
    <string name="item3_1">Opción 3- Submenú 1 - Calculadora</string>
    <string name="item3_2">Opción 3- Submenú 2 - Saír</string>

</resources>
```

1.3.3.3 O código java da aplicación

```
package com.example.u5_01_menus;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
```

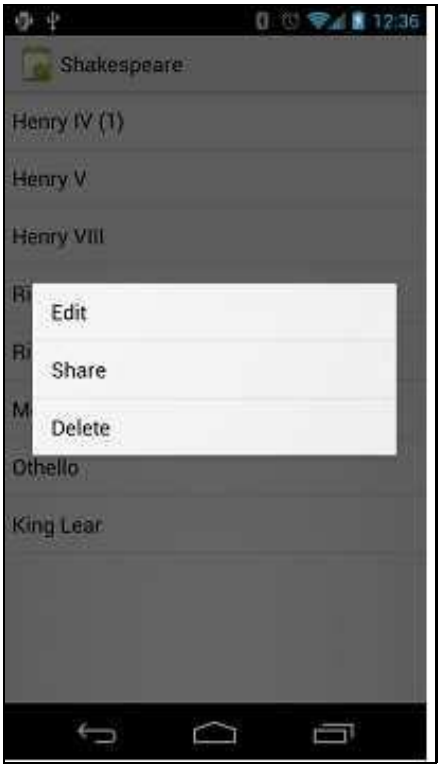
```
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView;
import android.widget.Toast;

public class U5_01_Menus extends Activity {

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_u5_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.u5_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.
/*
                int id = item.getItemId();
                if (id == R.id.action_settings) {
                        return true;
                }
                return super.onOptionsItemSelected(item);
        */
TextView tv = (TextView) findViewById(R.id.tv);
Toast.makeText(getApplicationContext(), "Pulsado elemento: " + item.getTitle().toString(), Toast.LENGTH_SHORT).show();

switch (item.getItemId()) {
case R.id.item1:
tv.setText("Quedarei con esta cara de sorriso cando remate esta tarefa");
return true;

case R.id.item2:
tv.setText("A opción 2 non ten nada asignado");
return true;

case R.id.SubItem3_1:
Intent intent = new Intent();
intent.setClassName("com.android.calculator2", "com.android.calculator2.Calculator");
startActivity(intent);
return true;

case R.id.SubItem3_2:
finish();
return true;
default:
return super.onOptionsItemSelected(item);

}
}

public void finish() {
super.finish();
}

}
```

- **Liña 39**: Recollemos o título do ítem pulsado.
- **Liñas 44,48,54,58**: devolver true no caso de procesar cada un dos ítems.
- **Liña 60**: Se non se procesou o ítem chamar ao pai, que vai devolver false por defecto.

- Se na definición de cada ítem se houbera usado o atributo **android:onClick** entón teríamos que crear en java os métodos correspondentes do mesmo xeito que se fai cando se usa ese atributo nun Layout.

1.4 Menús contextuais

- Pódense crear sobre calquera elemento View, pero normalmente vanse usar con ListView e GridView.*
- Existen dúas formas de implantar este tipo de menús:
- **Menú contextual flotante**:
- O menú aparece cando o usuario prema durante un tempo longo un elemento e aparece unha lista.



Para crear este tipo de menú:

- O View que queira usalo ten que ser asociado ao mesmo chamando o método **registerForContextMenu** pasando o obxecto View.
- Se todos os elementos do ListView / GridView teñen o mesmo menú contextual, se pode pasar como parámetro o ListView / GridView.
- Por exemplo (isto se fai no método onCreate): **registerForContextMenu(lista);**
    - ♦ Sendo lista a referencia a unha ListView.

- Implantar o método **onCreateContextMenu()** na Activity / Fragment.
    - ♦ Este método será chamado de forma automática cando o usuario preme durante un tempo o elemento (View) asociado o ContextMenu.
    - ♦ O que fai este método é amosar o menú contextual creado por nós previamente.

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.context_menu, menu);
}
```

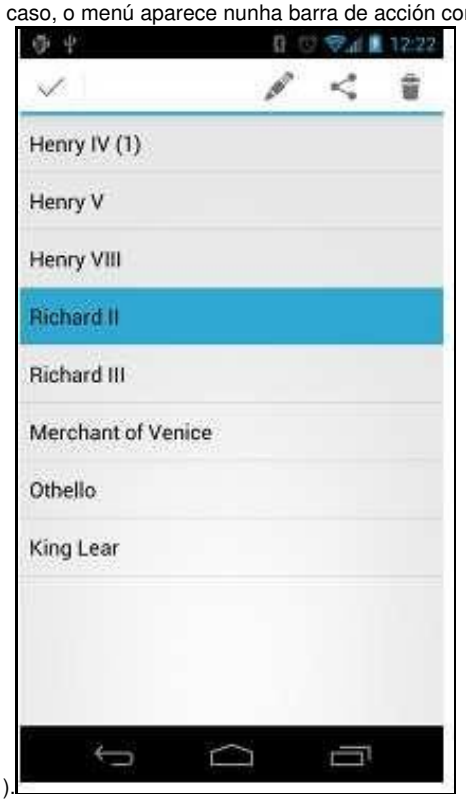
- Lembrar que o MenuInflater serve para pasar dun arquivo XML (o menú contextual definido por nós) a obxectos MenuItems.

- Implantar o método **onContextItemSelected()** ao que se chama de forma automática cando se selecciona algo do menú contextual

```
@Override
public boolean onContextItemSelected(MenuItem item) {
AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo();
switch (item.getItemId()) {
case R.id.edit:
editNote(info.id);
return true;
case R.id.delete:
deleteNote(info.id);
return true;
default:
return super.onContextItemSelected(item);
}
}
```

- **Liña 3**. Do ítem pulsado recibimos información extra, entre outras, o "id" e a "posición" do ítem do adaptador sobre o que se pulsou durante un tempo para crear o menú contextual.
    - Ese "id" e "posición" son recollidos na variable info de tipo AdapterContextMenuInfo.
- **Liña 4**: fixarse que é **item.getItemId()** para obter referencia ao id do menú contextual que foi seleccionado. Non confundir co id do elemento do adaptador sobre o que foi creado ese menú contextual.
- O resto funciona igual que non menú básico.

- **Menú de modo acción contextual**:
- Neste caso, o menú aparece nunha barra de acción contextual na parte superior da pantalla (só dispoñible a partires da versión 3.0 API11).



- O funcionamento é semellante ao anterior e deixamos para o participante no curso o seguinte enlace para que afonde no seu funcionamento:
    - http://developer.android.com/guide/topics/ui/menus.html#CAB

1.4.1 Menú contextual: Caso Práctico

- Neste caso vanse crear dous menús contextuais distintos:
    - un para unha etiqueta (TextView)
    - e outro para unha lista (ListView).

- Crear o proxecto: **U5_02_MenusContextuais**

- Creación dunha segunda Activity

- 

A aplicación ten unha etiqueta de texto (Hello world!) e unha lista.

- 

Se prememos durante un anaco sobre a etiqueta de texto aparece o menú contextual da imaxe. Se seleccionamos a primeira opción dese menú contextual ...

- 

... obtemos un Toast que nos di que seleccionamos esa opción.

- 

Se prememos por un anaco sobre un elemento da lista, por exemplo sobre Mango, obtemos outro menú contextual distinto. Se prememos na opción borrar ...

- 

... bórrase o elemento da lista

- 

Se prememos por un anaco de tempo sobre outro elemento, por exemplo Laranxa, e seleccionamos Duplicar do menú contextual ...

- 

... duplicamos o elemento na lista.

1.4.2 Menú contextual: O XML do Layout

- Conservamos o TextView que vén por defecto e engadimos un ListView.
- Ao TextView engadímoslle un id.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="@dimen/activity_vertical_margin" >

    <TextView
        android:id="@+id/tv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

    <ListView
        android:id="@+id/lvFroitas"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" >
    </ListView>

</LinearLayout>
```

1.4.3 Menú contextual: O XML dos menús contextuais

- Imos crear dous recursos xml para cada un dos menús contextuais distintos que temos segundo premamos sobre un TextView ou sobre un ListView.
- Poderían ser o mesmo menú contextual, pero así aprendemos que estes poden ser distintos en función da View ao que se asocien.

- **O xml de /res/menu/menu_contextual_etiqueta.xml**

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

    <item
        android:id="@+id/tvItem1"
        android:showAsAction="withText"
        android:title="TextView opción 1">
    </item>
    <item
        android:id="@+id/tvItem2"
        android:showAsAction="withText"
        android:title="TextView opción 2">
    </item>

</menu>
```

- **O xml de /res/menu/menu_contextual_lista.xml**

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

    <item
        android:id="@+id/lvItemBorrar"
        android:showAsAction="withText"
        android:title="Borrar">
    </item>
    <item
        android:id="@+id/lvItemDuplicar"
        android:showAsAction="withText"
        android:title="Duplicar">
    </item>
```

```
</menu>
```

- Observar que nos dous ficheiros xml cada item ten asociado un "id", para logo ser identificado o ítem cando sexa procesado no código.

1.4.4 Menú contextual: o código Java da aplicación

```
package com.example.u5_02_menuscontextuais;

import java.util.ArrayList;
import java.util.Arrays;
import android.app.Activity;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.ContextMenu.ContextMenuInfo;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView.AdapterContextMenuInfo;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class U5_02_MenusContextuais extends Activity {
ListView lv;
TextView tv;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_u5_02__menus_contextuais);

lv = (ListView) findViewById(R.id.lvFroitas);
tv = (TextView) findViewById(R.id.tv);
registerForContextMenu(tv);

engadirDatosListView();

}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.u5_02__menus_contextuais, 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.
int id = item.getItemId();
if (id == R.id.action_settings) {
return true;
}
return super.onOptionsItemSelected(item);
}

private void engadirDatosListView() {
String[] froitas = new String[] { "Laranxa", "Mango", "Ameixa" };

ArrayList<String> alFroitas = new ArrayList<String>();
```

```
alFroitas.addAll(Arrays.asList(froitas));

ArrayAdapter<String> adaptador = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, alFroitas);

adaptador.setDropDownViewResource(android.R.layout.simple_list_item_1);

lv.setAdapter(adaptador);

registerForContextMenu(lv);
}

@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo menuInfo) {
super.onCreateContextMenu(menu, v, menuInfo);
MenuInflater inflater = getMenuInflater();

// Comprobamos se o menú contextual se lanzou sobre a etiqueta ou sobre
// a lista
if (v.getId() == R.id.tv)
inflater.inflate(R.menu.menu_contextual_etiqueta, menu);

else if (v.getId() == R.id.lvFroitas)
inflater.inflate(R.menu.menu_contextual_lista, menu);
}

@Override
public boolean onContextItemSelected(MenuItem item) {
AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo();
ArrayAdapter<String> adaptador = (ArrayAdapter<String>) lv.getAdapter();

switch (item.getItemId()) {

// Ítems premidos sobre o TextView
// Lanza un Toast coa opción do menú contextual que se seleccinou
case R.id.tvItem1:
Toast.makeText(this, "Menú contextual TextView:\n"+item.getTitle(), Toast.LENGTH_SHORT).show();
return true;

case R.id.tvItem2:
Toast.makeText(this, "Menú contextual TextView:\n"+item.getTitle(), Toast.LENGTH_SHORT).show();
return true;

// Ítems premidos sobre o ListView
case R.id.lvItemBorrar:
adaptador.remove(adaptador.getItem(info.position));
adaptador.setNotifyOnChange(true);
return true;

case R.id.lvItemDuplicar:
adaptador.add(adaptador.getItem(info.position));
adaptador.setNotifyOnChange(true);

return true;
default:
return super.onContextItemSelected(item);
}
}

}
```

- **Liña 30**: indicamos que TextView vai ter un menú contextual asociado.

- **Liñas 55-69**: Creamos un Array estático que logo asignamos a un dinámico.
    - **Liña 62**: usamos o array dinámico para logo en tempo de execución poder engadir/borrar elementos do adaptador e por tanto do ListView.
    - **Liña 68**: indica que o ListView de froitas vai ter un menú contextual asociado.

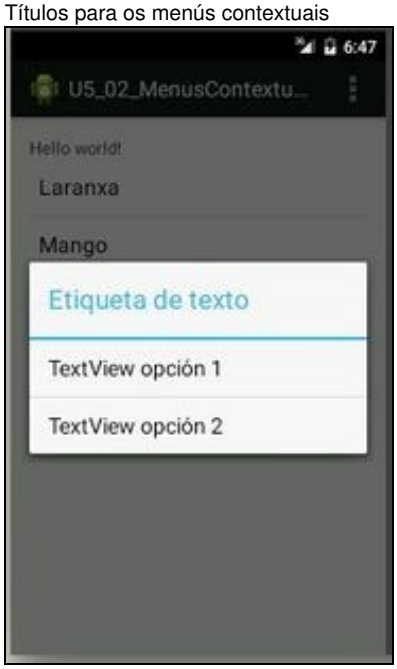
- **Liñas 71-83**: **onCreateContexMenu()** vai ser chamado cando se prema durante un anaco sobre unha view, neste caso comprobamos sobre que view se premeu e lánzase ("inflase") o correspondente menú contextual, cada un deles definido nun ficheiro xml distinto.

- **Liñas 85-116**: En función do id do ítem premido realizamos unhas accións ou outras como no caso dos menús básicos. Só que neste caso os ids proveñen de ítems de menús declarados en 2 ficheiros distintos.

1.4.4.1 Personalizar título do menú contextual

- Se queremos que o menú contextual teña un título debemos usar o método: **setHeaderTitle()**

- As seguintes imaxes amosan un exemplo para o caso do TextView e o ListView:

- Títulos para os menús contextuais
- 

Neste caso o título do menú contextual é "Etiqueta de texto"
- 

Neste outro é o título do ítem do ListView sobre o que se premeu.

- Os cambios a realizar no **código Java** son os seguintes:
- No memento no que hai que crear o menú contextual:

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo menuInfo) {
super.onCreateContextMenu(menu, v, menuInfo);
MenuInflater inflater = getMenuInflater();

// Comprobamos se o menú contextual se lanzou sobre a etiqueta ou sobre
// a lista
if (v.getId() == R.id.tv){
menu.setHeaderTitle("Etiqueta de texto");
inflater.inflate(R.menu.menu_contextual_etiqueta, menu);
}
else if (v.getId() == R.id.lvFroitas) {
AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo) menuInfo;
menu.setHeaderTitle(lv.getAdapter().getItem(info.position).toString());
inflater.inflate(R.menu.menu_contextual_lista, menu);
}
}
```

- **Liñas 9,14**: establecemos o título para o menú contextual.

-- Ángel D. Fernández González e Carlos Carrión Álvarez -- (2015).