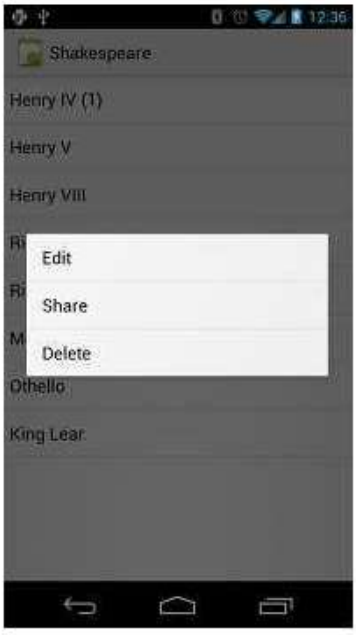
INDICE

3. Menús contextuales

- ✓ Se puede crear sobre cualquier elemento View, pero normalmente se van a usar con ListView y GridView.*
- ✓ Existen dos formas de implantar este tipo de menús:
- ✓ **Menú contextual flotante**:
- ✓ El menú aparece cuando el usuario pulsa durante un tiempo largo un elemento y aparece una lista.



Para crear este tipo de menú:

- ✓ El View que quiera usarlo tiene que ser asociado al mismo, llamando el método **registerForContextMenu** pasando el objeto View.
- ✓ Si todos los elementos del ListView / GridView tienen el mismo menú contextual, se puede pasar como parámetro el ListView / GridView.
- ✓ Por ejemplo (esto se hace en el método onCreate): **registerForContextMenu(lista);**
  - Siendo lista la referencia a una ListView.
- ✓ Implantar el método **onCreateContextMenu()** en la Activity / Fragment.
  - Este método será llamado de forma automática cuando el usuario pulse durante un tiempo el elemento (View) asociado al ContextMenu.
  - Lo que hace este método es mostrar el menú contextual creado por nosotros previamente.

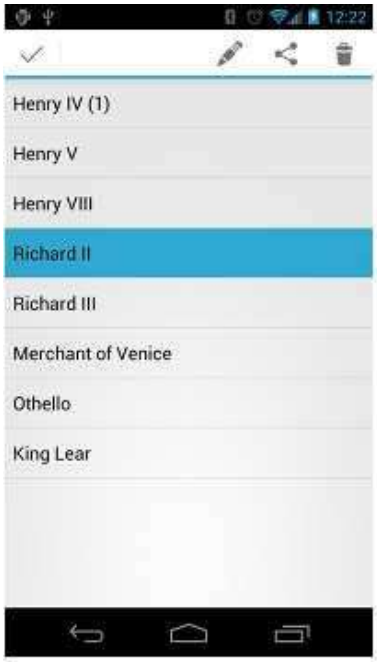
```
@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.context_menu, menu);
}
```

- ✓ Recordar que el MenuInflater sirve para pasar de un archivo XML (el menú contextual definido por nosotros) a objetos MenuItems.
- ✓ Fijarse que uno de los parámetros es el view sobre el que queremos mostrar el menú contextual. Si tenemos varios registrados (registerForContextMenu) nos servirá para saber de cual venimos.
- ✓ Implantar el método **onContextItemSelected()** al que se llama de forma automática cuando se selecciona algo del menú contextual

```
@Override
      public boolean onContextItemSelected(MenuItem item) {
               AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo();
               switch (item.getItemId()) {
               case R.id.edit:
          // editNote(info.id); Lanzaríamos una activity obteniendo el texto del elemento de
          la lista a partir de su id o con su posicición (info.position)
                        return true;
               case R.id.delete:
          // deleteNote(info.id); Borraríamos el elemento de la lista en base a su id o con su
          posicición (info.position)
                        return true;
               default:
                        return super.onContextItemSelected(item);
               }
      }
```

- ✓ **Línea 3**. Del ítem pulsado recibimos información extra, entre otras, el "id" y la "posición" del ítem del adaptador sobre el que se pulso durante un tiempo para crear el menú contextual.
  - Ese "id" y "posición" son recogidos en la variable info de tipo AdapterContextMenuInfo.

- ✓ **Línea 4:** fijarse que é **item.getItemId()** para obtener referencia al id del menú contextual que fue seleccionado. No confundir con id del elemento del adaptador sobre el que fue creado ese menú contextual.
- ✓ El resto funciona igual que en el menú básico.
- ✓ **Menú de modo acción contextual**:
- ✓ En este caso, el menú aparece en una barra de acción contextual en la parte superior de la pantalla (solo disponible a partir de la versión 3.0 API11).



- ✓ El funcionamiento es semejante al anterior. Dejamos para el estudiante el siguiente enlace para ahondar en su funcionamiento: http://developer.android.com/guide/topics/ui/menus.html#CAB

3.1. Menú contextual: Caso Práctico aplicado a dos Views

- ✓ Dentro del paquete **Menus** crear una nueva 'Empty Activity' de nombre: **UD06_02_Menus** de tipo Launcher.
- ✓ Modificar el archivo AndroidManifiest.xml y añade una label a la activity.
- ✓ En este ejemplo vamos a asociar dos menús contextuales diferentes a dos views cualquiera. En el ejemplo serán dos ImageView, pero se podría emplear cualquier View.

- ✓ El proceso que tenéis que aprender es el siguiente:
  - Primer paso, se deben crear el/los menús emergentes que queramos que aparezcan (las opciones de menú).
  - Segundo paso, se debe asociar un Context Menu a cada view. Para eso se tiene que llamar al método registerForContextMenu() y pasarle como parámetro el view que queramos que tenga un Context Menu.
  - Tercer paso, se debe sobreescribir el método onCreateContextMenu(). A este método llegarán todos los views que estén registrados en el paso anterior. Es aquí donde tenemos que hacer la operación de 'inflater' del menú asociado.
  - Cuarto paso, se debe sobreescribir el método onContextItemSelected() donde llegará la ejecución al pulsar sobre uno de los menú items del menú emergente.

- ✓ Veamos estos cuatro pasos aplicados a nuestro ejemplo:

  Primer paso: Creamos los menús emergentes que van aparecer. En el ejemplo creamos dos.

Archivo: /res/menu/menu_context1_ud06_02_menus.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/mniAdd_UD06_02_Menus"
        android:title="Añadir" />
</menu>
```

Archivo: /res/menu/menu_context2_ud06_02_menus.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/mniDel_UD06_02_Menus"
        android:title="Borrar" />
</menu>
```

Segundo Paso: Asociar un Context Menu a cada view.

Activity: UD06_02_Menus.java

```
package com.example.ud06_01_menus;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Toast;

public class UD06_02_Menus extends AppCompatActivity {

    private void asociarMenusContextual(){

        registerForContextMenu(findViewById(R.id.imvEstrella_UD06_02_Menus));
        registerForContextMenu(findViewById(R.id.imvCrono_UD06_02_Menus));

     }
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_u_d06_02__menus);

        asociarMenusContextual();
    }
}
```

Tercer Paso: Se debe sobreescribir el método onCreateContextMenu()

Activity: UD06_02_Menus.java

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

    MenuInflater inflater = getMenuInflater();
    switch(v.getId()) {
        case R.id.imvEstrela_UD06_02_Menus:
            inflater.inflate(R.menu.menu_context1_ud06_02_menus,menu);
            break;
        case R.id.imvCrono_UD06_02_Menus:
            inflater.inflate(R.menu.menu_context2_ud06_02_menus,menu);
            break;
    }
}
```

Línea 6: En función del view que va a activar Menú Contextual,...

Línea 8,11: Cargamos un menú u otro.

Cuarto Paso: Se debe sobreescribir el método onContextItemSelected()

Activity: UD06_02_Menus.java

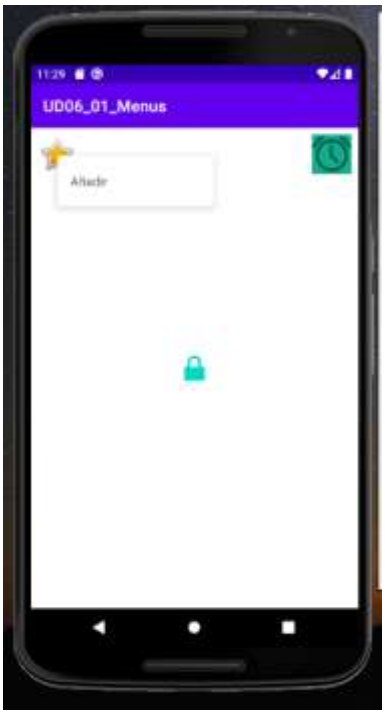
```
@Override
public boolean onContextItemSelected(MenuItem item) {
    switch(item.getItemId()) {
        case R.id.mniAdd_UD06_02_Menus:
             Toast.makeText(this,"Pulsada la opción Añadir en la ima-
```

```
gen",Toast.LENGTH_SHORT).show();
                break;
            case R.id.mniDel_UD06_02_Menus:
                Toast.makeText(this,"Pulsada la opción Borrar en la ima-
gen",Toast.LENGTH_SHORT).show();
                break;
        }
        return super.onContextItemSelected(item);
}
```

En función de la opción escogida del menú contextual haremos una acción concreta.

3.1.1. Menú popup: Caso Práctico aplicado a dos Views

✓ Existe otra forma de 'emular' un menú emergente y es empleando un evento LongClick y haciendo aparecer un PopupMenu.



Vamos a modificar el ejercicio anterior y vamos a añadir una nueva imagen, en el que aparecerá el mismo menú emergente que empleamos en la estrella.

```
<?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_02_Menus">

    <ImageView
        android:id="@+id/imvEstrella_UD06_02_Menus"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
```

```
        android:contentDescription="Estrella"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@android:drawable/btn_star_big_on" />

    <ImageView
        android:id="@+id/imvCrono_UD06_02_Menus"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:background="#10AC8C"
        android:contentDescription="Reloj"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@android:drawable/ic_lock_idle_alarm" />
    <ImageView
        android:id="@+id/imvCandado_UD06_02_Menus"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@android:drawable/ic_lock_lock" />

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

✓ Ahora vamos a modificar el código de la Activity para que muestre el PopUp. Recordar que no creamos un menu nuevo ya que vamos a emplear el definido para la estrella.

Lo que tenemos que hacer es capturar el evento LongClick sobre la imagen y crear un objeto de la clase PopupMenu.

Activity UD06_02_Menus.java

```
private void asociarMenuPopup(){
    findViewById(R.id.imvCandado_UD06_02_Menus).setOnLongClickListener(
            new View.OnLongClickListener() {
                @Override
                public boolean onLongClick(View v) {
                    PopupMenu popupMenu = new PopupMenu(getApplicationContext(),v);
                    popupMenu.inflate(R.menu.menu_context1_ud06_02_menus);
                    popupMenu.show();

                    popupMenu.setOnMenuItemClickListener(new an-
droid.widget.PopupMenu.OnMenuItemClickListener() {
                        @Override
                        public boolean onMenuItemClick(MenuItem menuItem) {
                            if (menuItem.getItemId()==R.id.mniAdd_UD06_02_Menus) {
                                Toast.makeText(getApplicationContext(), "Pulsado
",Toast.LENGTH_SHORT).show();
                                return true;
                            }
                            return false;
                        }
                    });
                    return true;
                }
            }
    );
}

@Override
```

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_u_d06_02__menus);

    asociarMenusContextual();
    asociarMenuPopup();
}
```

- ✓ Para crear un objeto PopupMenu debemos:
  - Instanciarlo (crear un objeto de la clase PopupMenu) pasando como parámetros al constructor una referencia al contexto de la activity y el view sobre el que se va a generar el popupmenu (en nuestro caso la imagen): **`PopupMenu popupMenu = new PopupMenu(getApplicationContext(),v);`**
  - Inflar el menú llamando al método inflate de la instancia anterior y pasando como parámetro la referencia al /res/menu/ que queramos que aparezca: popupMenu.inflate(R.menu.menu_context1_06_02_menus);
  - Llamar al método show(): popupMenu.show();
- ✓ Ahora para gestionar los eventos de 'Click' sobre el PopupMenu, debemos hacer uso do método setOnMenuItemClickListener() del PopupMenu.

  Al igual que cualquier evento, dicho método espera recibir un objeto de una clase que implementa la interface 'android.widget.PopupMenu.OnMenuItemClickListener'. Podemos hacerlo a nivel de Activity o una clase anónima como cualquier otro evento:

Activity UD06_02_Menus.java

```
private void asociarMenuPopup(){
    findViewById(R.id.imvCandado_UD06_02_Menus).setOnLongClickListener(
            new View.OnLongClickListener() {
                @Override
                public boolean onLongClick(View v) {
                    PopupMenu popupMenu = new PopupMenu(getApplicationContext(),v);
                    popupMenu.inflate(R.menu.menu_context1_ud06_02_menus);
                    popupMenu.show();

                    popupMenu.setOnMenuItemClickListener(new android.widget.PopupMenu.OnMenuItemClickListener() {
                        @Override
                        public boolean onMenuItemClick(MenuItem menuItem) {
                            if (menuItem.getItemId()==R.id.mniAdd_UD06_02_Menus) {
                                Toast.makeText(getApplicationContext(), "Pulsado",Toast.LENGTH_SHORT).show();
                                return true;
                            }
                            return false;
                        }
                    });
                    return true;
                }
            }
    );}
```

3.2. Menú contextual: Caso Práctico aplicado a un RecyclerView

- ✓ Partimos que ya tenemos hecho estudiado el PDF sobre RecyclerView y tenemos la actividad siguiente:

✓ Este sería un caso típico donde queremos que cuando mantengamos pulsado un elemento da lista, aparezca un menú emergente y podamos escoger la opción 'Eliminar'.

3.2.1. Opción 1) Registrar un ContextMenu

✓ El problema que vamos a tener es que donde se encuentran los datos es en la Activity, mientras que donde se encuentran los componentes gráficos donde queremos asociar un ContextMenu se encuentran en el ViewHolder.

✓ Según vimos anteriormente, para registrar un ContextMenu tenemos que llamar al método registerForContextMenu() y pasarle como parámetro cada elemento de la lista.

Cada elemento de la lista se encuentra en la clase ViewHolder creada por nosotros, concretamente en el constructor:

```
package com.example.ud06_01_menus;

import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;


public class ViewHolder_UD06_03_RecycleViewCardView extends RecyclerView.ViewHolder {
    public ImageView itemImagen;
    public TextView itemTexto;


    public ViewHolder_UD06_03_RecycleViewCardView(@NonNull final View itemView) {
        super(itemView);

        itemImagen = itemView.findViewById(R.id.imgImaxe_UD06_03_CardLayout);
        itemTexto = itemView.findViewById(R.id.tvTexto_UD06_03_CardView);

        ((UD06_03_Menus)itemView.getContext()).registerForContextMenu(itemView);
```

```
        itemImagen.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Planetas_UD06_03_RecycleViewCardView planetaSeleccionado = (Plane-
tas_UD06_03_RecycleViewCardView) v.getTag();
                Toast.makeText(v.getContext(), "Pulsado elemento " + getAdapterPosition() + " de
la lista.", Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

El objeto 'itemview' representa todos los views que conforman cada elemento de la lista (en nuestro ejemplo, la imagen y el textview).

No podemos llamar a registerContextView ya que dicho método pertenece a la clase Activity, por lo que es necesario obtener una referencia a dicha clase.

Lo hacemos a través de la clase Context, ya que podemos obtener una referencia al contexto a partir del 'itemview'.

✓ Una vez hecho hacemos como en el ejercicio anterior (estamos empleando el menú contextual hecho en el ejercicio anterior, podéis crear uno nuevo si queréis):

Activity: UD06_03_Menus

```
package com.example.ud06_01_menus;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.os.Bundle;
import android.view.ContextMenu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Toast;

import com.google.android.material.floatingactionbutton.FloatingActionButton;

import java.util.ArrayList;

public class UD06_03_Menus extends AppCompatActivity {

    private RecycleViewAdapter_UD06_06_RecycleViewCardView recycleAdapter;

    private ArrayList<Planetas_UD06_03_RecycleViewCardView> planetas;

    @Override
    public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo me-
nuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);

        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.menu_context2_ud06_02_menus,menu);

    }
```

```
    @Override
    public boolean onContextItemSelected(MenuItem item) {

        if (item.getItemId()==R.id.mniDel_UD06_02_Menus){
            Toast.makeText(this,"Debemos eliminar este elemento de
menú",Toast.LENGTH_SHORT).show();
        }

        return super.onContextItemSelected(item);
    }


    private void inicializarDatosPlanetas(){

        planetas = new ArrayList<>();

        planetas.add(new Plane-
tas_UD06_03_RecycleViewCardView(1,"MERCURIO",false,R.drawable.mercurio));
        planetas.add(new Plane-
tas_UD06_03_RecycleViewCardView(2,"VENUS",false,R.drawable.venus));
        planetas.add(new Plane-
tas_UD06_03_RecycleViewCardView(3,"TIERRA",false,R.drawable.tierra));
        planetas.add(new Plane-
tas_UD06_03_RecycleViewCardView(4,"JUPITER",false,R.drawable.jupiter));
        planetas.add(new Plane-
tas_UD06_03_RecycleViewCardView(5,"SATURNO",false,R.drawable.saturno));

    }

    private void inicializarRecycleView(){

        recycleAdapter = new RecycleViewAdapter_UD06_06_RecycleViewCardView(planetas);

        RecyclerView.LayoutManager layoutManager = new LinearLayoutManager(this);
        RecyclerView recyclerView = findViewById(R.id.rvwRecycleView);
        recyclerView.setLayoutManager(layoutManager);
        recyclerView.setAdapter(recycleAdapter);


    }

    private void xestionarEventos(){

        FloatingActionButton fab = findViewById(R.id.fabAdd_ud04_01_recyclecardview);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                planetas.add(new Plane-
tas_UD06_03_RecycleViewCardView(6,"URANO",false,R.drawable.urano));
                recycleAdapter.notifyItemInserted(planetas.size());
            }
        });

    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_u_d06_03__menus);

        inicializarDatosPlanetas();
        inicializarRecycleView();
        xestionarEventos();

    }

}
```

✓ Ahora el problema es saber que elemento de la lista necesitamos borrar.

Al método 'onContextItemSelected()' llega el elemento del menú seleccionado, pero nada acerca del elemento de la lista.

Podemos aprovechar que el ItemView seleccionado llega al método onCreateContextMenu() y recoger ahí cuál es la view sobre la que estamos generando el menú contextual.

Ya vimos en el ImageView y también en el RecyclerView como asociar información a un view: Empleando la propiedad Tag.

El tag ya lo tenemos 'cogido' y guardamos en él el objeto planeta del ejercicio hecho en el punto RecyclerView:

Archivo: RecycleViewAdapter_UD06_03_RecycleViewCardView

```
@Override
public void onBindViewHolder(@NonNull RecyclerView.ViewHolder viewHolder, int i) {
    ViewHolder_UD06_03_RecycleViewCardView viewHolderMeu = (ViewHol-
der_UD06_03_RecycleViewCardView) viewHolder;
    viewHolderMeu.itemView.setTag(planetas.get(i));
    viewHolderMeu.itemImagen.setImageResource(planetas.get(i).getFotoId());
    viewHolderMeu.itemTexto.setText(planetas.get(i).getNombre());
}
```

Podemos por tanto guardar esta información en la activity para después ser empleada en el método onContextItemSelected():

Archivo: UD06_01_RecycleViewCardView

```
.....
public class UD06_03_Menus extends AppCompatActivity {

    private RecycleViewAdapter_UD06_03_RecycleViewCardView recycleAdapter;

    private ArrayList<Planetas_UD06_03_RecycleViewCardView> planetas;

    private Planetas_UD06_03_RecycleViewCardView planetaSeleccionado;

    @Override
    public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo me-
nuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);

        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.menu_context2_ud06_02_menus,menu);

        planetaSeleccionado = (Planetas_UD06_03_RecycleViewCardView)  v.getTag();
    }

    @Override
    public boolean onContextItemSelected(MenuItem item) {

        if (item.getItemId()==R.id.mniDel_UD06_02_Menus){
            Toast.makeText(this,"Debemos eliminar este elemento de
```

```
menú",Toast.LENGTH_SHORT).show();
                planetas.remove(planetaSeleccionado);
                recycleAdapter.notifyDataSetChanged();
            }

            return super.onContextItemSelected(item);
        }
……..
```

- **Línea 27**: Fijarse que tenemos que llamar a notifyDataSetChanged() ya que no sabemos cuál es el índice del elemento seleccionado.

✓ Podemos 'optimizar' y hacer que solamente 'actualice' la lista sobre el elemento borrado:

```
@Override
public boolean onContextItemSelected(MenuItem item) {

    if (item.getItemId()==R.id.mniDel_UD06_02_Menus){
        Toast.makeText(this,"Debemos eliminar este elemento de menú",Toast.LENGTH_SHORT).show();
        int indicePlaneta = planetas.indexOf(planetaSeleccionado);
        planetas.remove(planetaSeleccionado);
        recycleAdapter.notifyItemRemoved(indicePlaneta);
    }

    return super.onContextItemSelected(item);
}
```

3.2.2. Opción 2) Emplear un PopupMenu

✓ Te animas a hacerlo?

Como pista comentar que hay que registrar la interface OnLongClickListener en el itemView del ViewHolder, pero tendréis que informarle que dicha interface esté implementada en la Activity.

Implementar la interface en la Activity y seguir como en el ejemplo anterior del PopupMenu.

3.3. Menú contextual: Caso Práctico aplicado a una ListView y un TextView

Si no tenemos creado el paquete **Menus**, crearemos un nuevo paquete de nombre: **Menus.**

✓ Dentro del paquete **Menus** crear una nueva 'Empty Activity' de nombre: **UD06_04_Menus** de tipo Launcher.

✓ Modificar el archivo **AndroidManifiest.xml** y añadir una label a la activity.

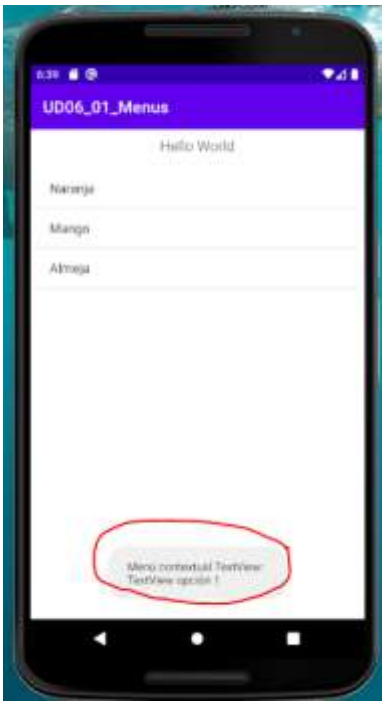
✓ En este caso se van a crear dos menús contextuales distintos:

- Un para una etiqueta (TextView)
- Y otro para una lista (ListView)

Creando menús contextuales

La aplicación tiene una etiqueta de texto (Hello world!) y una lista.

El código generado. Si pulsamos durante un rato sobre la etiqueta de texto aparece el menú contextual de la imagen. Si seleccionamos la primera opción de ese menú contextual ...



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

…Se borra el elemento de la lista…

Se pulsamos un rato sobre un elemento de la lista, por ejemplo sobre Mango, obtenemos otro menú contextual distinto. Si pulsamos en la opción borrar ...

Si pulsamos por un rato sobre otro elemento, por ejemplo Naranja, y seleccionamos Duplicar del menú contextual ...

... duplicamos el elemento en la lista.

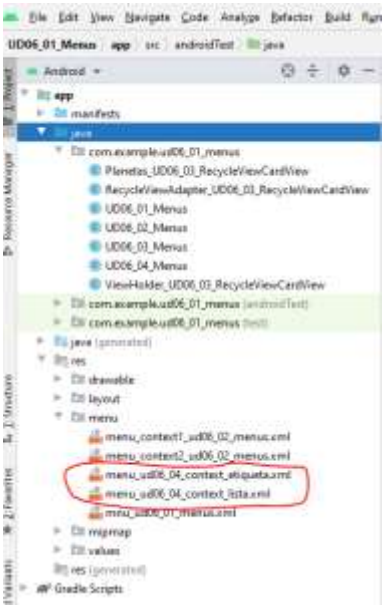
3.3.1. Menú contextual: El XML del Layout

```
<?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_04_Menus">


    <TextView
        android:id="@+id/tvTexto_UD06_04_Menus"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Hello World"
        android:textSize="18sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ListView
        android:id="@+id/lvFroitas_UD06_04_Menus"
        android:layout_width="0dp"
        android:layout_height="169dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
```

```
        android:layout_marginTop="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/tvTexto_UD06_04_Menus" />

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

3.3.2. Menú contextual: El XML de los menús contextuales

- ✓ Vamos a crear dos recursos xml para cada uno de los menús contextuales distintos que tenemos según pulsemos sobre un TextView o sobre un ListView.
- ✓ Podrían ser el mismo menú contextual, pero así aprendemos que estos pueden ser distintos en función de la View al que se asocien.



El xml de /res/menu/menu_ud06_02_context_etiqueta.xml

Nota: El Android Studio seguramente marcará un error en la línea **android:showAsAction** y que deberéis cambiarlo por 'app:showAsAction'. Ese aviso aparece debido a que tenemos importada la librería de compatibilidad v7 en alguna otra práctica. Solamente sería necesario si la activity derivase de la clase AppCompactActivity como hicimos en la práctica anterior (deberéis de aplicarlo a vuestro caso).

```
<?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_04_context_etiq"
        app:showAsAction="withText"
        android:title="TextView opción 1">
    </item>
    <item
        android:id="@+id/mniItem2_04_04_context_etiq"
        app:showAsAction="withText"
        android:title="TextView opción 2">
```

```
    </item>
</menu>
```

El xml de /res/menu/menu_ud06_02_context_lista.xml

Nota: El Android Studio seguramente marcará un error en la línea **android:showAsAction** y que deberéis cambiarlo por 'app:showAsAction'. No le hagáis caso. Ese aviso aparece debido a que tenemos importada la librería de compatibilidad v7 en alguna otra práctica. Solamente sería necesario si la activity derivase de la clase AppCompactActivity como hicimos en la práctica anterior.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/mniItemBorrar_ud06_04_context_lista"
        android:showAsAction="withText"
        android:title="Borrar">
    </item>
    <item
        android:id="@+id/mniItemDuplicar_ud06_04_context_lista"
        android:showAsAction="withText"
        android:title="Duplicar">
    </item>
</menu>
```

✓ Observar que en los dos ficheros xml cada item tiene asociado un "id", para luego ser identificado el ítem cuando sea procesado en el código.

3.3.3. Menú contextual: el código Java de la aplicación

```
package com.example.ud06_01_menus;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.ContextMenu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Arrays;

public class UD06_04_Menus extends AppCompatActivity {

    private void registrarMenusEmergentes(){

        TextView tv = findViewById(R.id.tvTexto_UD06_04_Menus);
        registerForContextMenu(tv);

        ListView lv = findViewById(R.id.lvFroitas_UD06_04_Menus);
        registerForContextMenu(lv);
    }

    private void anadirDatosLista(){
        String[] frutas = new String[] { "Naranja", "Mango", "Almeja" };
        ArrayList<String> alFrutas = new ArrayList<String>();
```

```
        alFrutas.addAll(Arrays.asList(frutas));

        ArrayAdapter<String> adaptador = new ArrayAdapter<String>(this, an-
droid.R.layout.simple_list_item_1, alFrutas);
        adaptador.setDropDownViewResource(android.R.layout.simple_list_item_1);

        ListView lv = findViewById(R.id.lvFroitas_UD06_04_Menus);
        lv.setAdapter(adaptador);

    }
```

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

        // Comprobamos si el menú contextual se lanzó sobre la etiqueta o sobre la lista
        if (v.getId() == R.id.tvTexto_UD06_04_Menus) {
            inflater.inflate(R.menu.menu_ud06_04_context_etiqueta, menu);
        }

        else if (v.getId() == R.id.lvFroitas_UD06_04_Menus) {
            inflater.inflate(R.menu.menu_ud06_04_context_lista, menu);
        }
    }
```

```
    @Override
    public boolean onContextItemSelected(MenuItem item) {
        AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo)
item.getMenuInfo();

        ListView lv = findViewById(R.id.lvFroitas_UD06_04_Menus);
        ArrayAdapter<String> adaptador = (ArrayAdapter<String>) lv.getAdapter();

        switch (item.getItemId()) {

            // Ítems pulsados sobre el TextView
            // Lanza un Toast con la opción del menú contextual que se selecciono
            case R.id.mniItem1_04_04_context_etiq:
                Toast.makeText(this, "Menú contextual TextView:\n"+item.getTitle(), To-
ast.LENGTH_SHORT).show();
                return true;

            case R.id.mniItem2_04_04_context_etiq:
                Toast.makeText(this, "Menú contextual TextView:\n"+item.getTitle(), To-
ast.LENGTH_SHORT).show();
                return true;

            // Ítems pulsados sobre el ListView
            case R.id.mniItemBorrar_ud06_04_context_lista:
                adaptador.remove(adaptador.getItem(info.position));
                adaptador.setNotifyOnChange(true);
                return true;
```

```
            case R.id.mniItemDuplicar_ud06_04_context_lista:
                adaptador.add(adaptador.getItem(info.position));
                adaptador.setNotifyOnChange(true);

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

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_u_d06_04__menus);

        registrarMenusEmergentes();
        anadirDatosLista();
    }
}
```

- ✓ **Línea 26:** indicamos que TextView va a tener un menú contextual asociado.
- ✓ **Línea 29:** indica que el ListView de frutas va a tener un menú contextual asociado.
- ✓ **Líneas 33,34:** Creamos un Array estático que luego asignamos a uno dinámico.
- ✓ **Línea 36:** usamos el array dinámico para luego en tiempo de ejecución poder añadir/borrar elementos del adaptador y por por tanto del ListView.

Nota: Se ponemos un array como parámetro en el ArrayAdapter, no podríamos hacer operaciones de borrado y añadir nuevos elementos.

- ✓ **Líneas 46-60: onCreateContexMenu()** va a ser llamado cuando durante un rato se pulse sobre la view, en este caso comprobamos sobre que view se pulso y se lanza ("se infla") el correspondiente menú contextual, cada uno de ellos definido en un fichero xml distinto.
- ✓ **Líneas 62-95:** En función del id del ítem pulsado realizamos unas acciones u otras como en el caso de los menús básicos. Solo que en este caso los ids provienen de ítems de menús declarados en 2 ficheros distintos.

3.3.4. Personalizar título del menú contextual

- ✓ Si queremos que el menú contextual tenga un título debemos usar el método: **setHeaderTitle()**
- ✓ Las siguientes imágenes muestran un ejemplo para el caso del TextView y el ListView:

En este caso el título del menú contextual es "Etiqueta de texto"

En este otro es el título del ítem del ListView sobre o que se pulso.

- ✓ Los cambios a realizar en el código Java son los siguientes:
- ✓ En el momento en el que hay que crear el menú contextual:

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

    // Comprobamos si el menú contextual se lanzó sobre la etiqueta o sobre la lista
    if (v.getId() == R.id.tvTexto_UD06_04_Menus) {
        menu.setHeaderTitle("Etiqueta de texto");
        inflater.inflate(R.menu.menu_ud06_04_context_etiqueta, menu);
    }

    else if (v.getId() == R.id.lvFroitas_UD06_04_Menus) {
        AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo) menuInfo;

        ListView lv = findViewById(R.id.lvFroitas_UD06_04_Menus);
        menu.setHeaderTitle(lv.getAdapter().getItem(info.position).toString());
        inflater.inflate(R.menu.menu_ud06_04_context_lista, menu);
    }
}
```