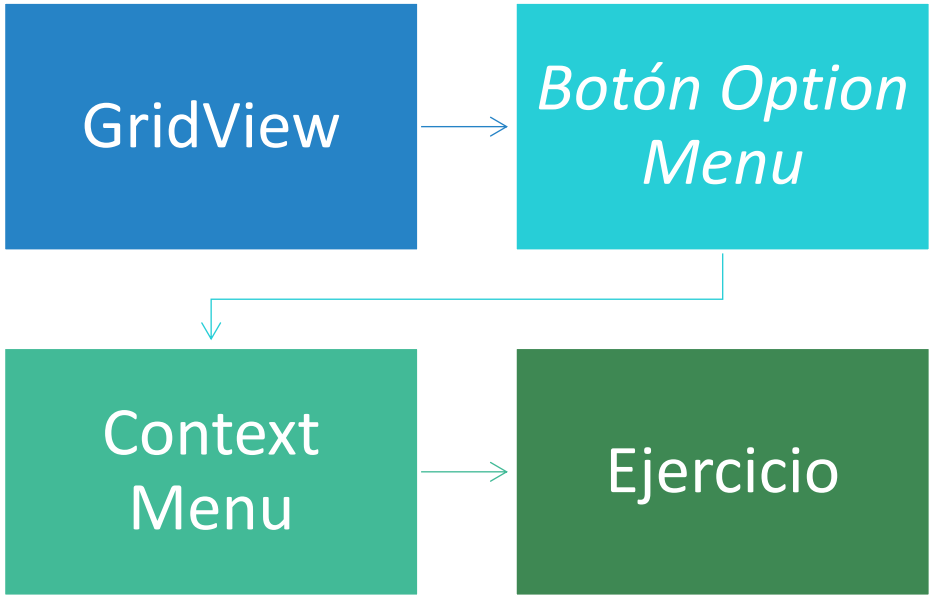
2º DAM

Programación Multimedia y Dispositivos Móviles

Programación Android

José A. Lara

Contenido



android

GridView

Programación Android

GridView

Un GridView es una vista que muestra los elementos en una cuadrícula desplazable en horizontal y en vertical.

Creamos un nuevo layout (grid_item) para los elementos que van a visualizarse en el GridView.

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="120dp">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_margin="25dp"
        android:src="@android:drawable/btn_star_big_on"/>

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#333333"
        android:layout_gravity="center_horizontal"
        android:text="" />
</LinearLayout>
```

android

Programación Android

GridView

Creamos un Layout con el GridView

```xml
<?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=".GridActivity">

    <GridView
        android:id="@+id/gridView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:columnWidth="90dp"
        android:gravity="center"
        android:horizontalSpacing="10dp"
        android:numColumns="auto_fit"
        android:stretchMode="columnWidth"
        android:verticalSpacing="10dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Programación Android

GridView

```
class MainActivity : AppCompatActivity() {
    private lateinit var nombres:MutableList<String>
    private lateinit var adapter: MyAdapter
    private var contador:Int=0
```

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    val gridView:GridView = findViewById(R.id.gridView)
    nombres=mutableListOf("Luis","Ruben","Jose","Santiago","Luisa","Roberto","Josefina","Maria","Lorena",
    "Renato","Juan","Sofia","Lara","Juliana","Manuel","Susana","Pedro","Pablo","Felipe","Antonia",
    "Ines","Sonia","Alberto","Paula","Gustavo","Alfonso","Irene")
    adapter = MyAdapter( context: this, R.layout.grid_item, nombres)
    gridView.adapter=adapter
    gridView.setOnItemClickListener { adapterView, view, position, id ->
        Toast.makeText(
            context: this,  text: "Has hecho click sobre " +
                    nombres.get(position), Toast.LENGTH_SHORT
        ).show()
    }
}
```

android

Programación Android

GridView

```
public class GridActivity extends AppCompatActivity {
    private List<String> nombres;
    private GridView gridView;
    private int contador=0;
    private MyAdapter adapter;
```

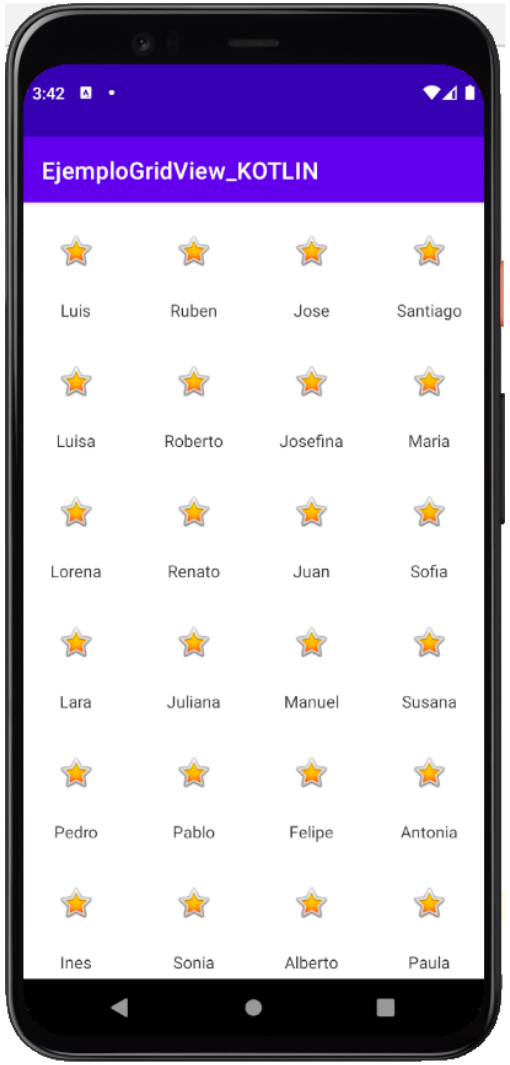
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_grid);
    gridView = findViewById(R.id.gridView);
    nombres = new ArrayList<String>();
    nombres.add("Luis"); nombres.add("Ruben"); nombres.add("Jose");
    nombres.add("Santiago"); nombres.add("Luisa"); nombres.add("Roberto");
    nombres.add("Josefina"); nombres.add("Maria"); nombres.add("Lorena");
    nombres.add("Renato"); nombres.add("Juan"); nombres.add("Sofia");
    nombres.add("Lara"); nombres.add("Juliana"); nombres.add("Manuel");
    nombres.add("Susana"); nombres.add("Pedro"); nombres.add("Pablo");
    nombres.add("Felipe"); nombres.add("Antonia"); nombres.add("Ines");
    nombres.add("Sonia"); nombres.add("Alberto"); nombres.add("Paula");
    nombres.add("Gustavo"); nombres.add("Alfonso"); nombres.add("Irene");
    gridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> adapterView, View view, int position, long id) {
            Toast.makeText( context: GridActivity.this,  text: "Has hecho click sobre "+
                            nombres.get(position),Toast.LENGTH_SHORT).show();
        }
    });
    adapter = new MyAdapter( context: this, R.layout.grid_item, nombres);
    gridView.setAdapter(adapter);
}
```

android

Programación Android

GridView

.



android

Botón Option Menu

Botón Option Menu

Este botón lo vamos a utilizar en este ejemplo para añadir nuevos elementos al GridView (o al ListView).
En primer lugar creamos un nuevo archivo de recursos (menu.xml) en la carpeta res/menu.

```
<?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/add_item"
        android:icon="@android:drawable/ic_menu_add"
        android:title="@string/add_elemento"
        app:showAsAction="ifRoom"
        />
</menu>
```

Botón Option Menu

Sobreescribimos los métodos onCreateOptionsMenu() para crear el menú y onOptionsItemSelected() para manejar los eventos en el menú.

```
override fun onCreateOptionsMenu(menu: Menu?): Boolean {
    val inflater = menuInflater
    inflater.inflate(R.menu.menu, menu)
    return true
}

override fun onOptionsItemSelected(item: MenuItem): Boolean {
    return when (item.itemId) {
        R.id.add_item -> {
            nombres.add("Nuevo n°" + ++contador)
            adapter.notifyDataSetChanged()
            true
        }
        else -> super.onOptionsItemSelected(item)
    }
}
```

Botón Option Menu

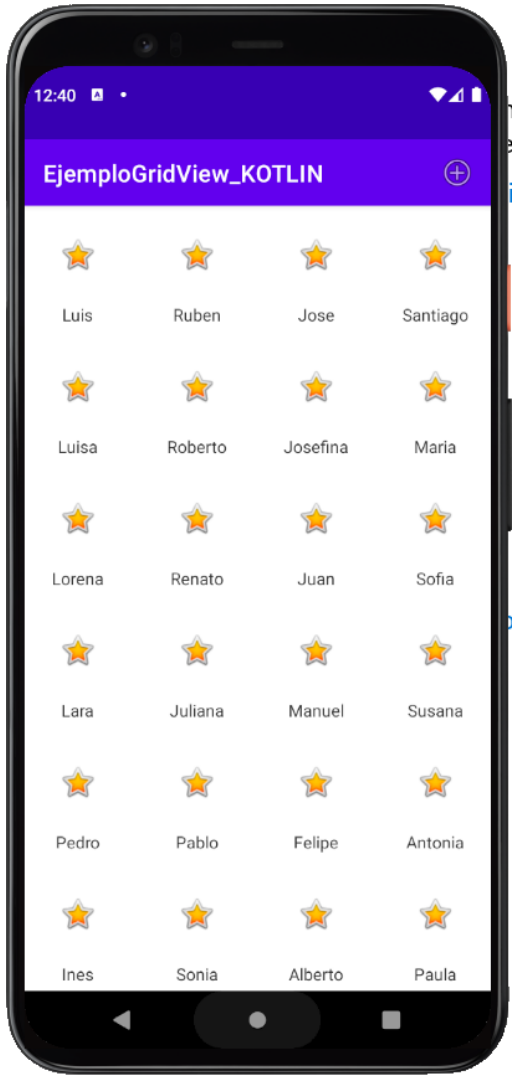
Sobreescribimos los métodos onCreateOptionsMenu() para crear el menú y onOptionsItemSelected() para manejar los eventos en el menú.

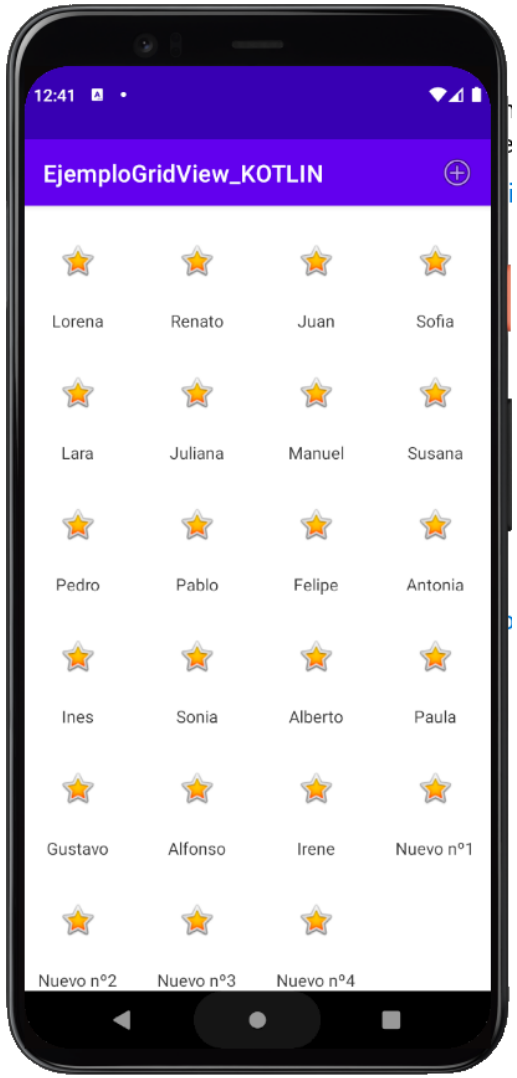
```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.menu, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    switch(item.getItemId()){
        case R.id.add_item:
            nombres.add("Nuevo n°"+(++contador));
            adapter.notifyDataSetChanged();
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

Programación Android

Botón Option Menu





android

Context Menu

Context Menu

Creamos un nuevo menú **context_menu.xml**

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

Context Menu

Implementamos dos métodos: **onCreateContextMenu()** para crear el menú contextual y **onContextItemSelected()** para manejar los eventos.

```
override fun onCreateContextMenu(menu: ContextMenu, v: View?, menuInfo: ContextMenuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo)
    val inflater = menuInflater
    val info = menuInfo as AdapterContextMenuInfo
    menu.setHeaderTitle(nombres[info.position])
    inflater.inflate(R.menu.context_menu, menu)
}

override fun onContextItemSelected(item: MenuItem): Boolean {
    val info = item.menuInfo as AdapterContextMenuInfo
    return when (item.itemId) {
        R.id.delete_item -> {
            nombres.removeAt(info.position)
            adapter.notifyDataSetChanged()
            true
        }
        else -> super.onContextItemSelected(item)
    }
}
```

Context Menu

Implementamos dos métodos: **onCreateContextMenu()** para crear el menú contextual y **onContextItemSelected()** para manejar los eventos.

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    MenuInflater inflater = getMenuInflater();
    AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo) menuInfo;
    menu.setHeaderTitle(nombres.get(info.position));
    inflater.inflate(R.menu.context_menu,menu);
}

@Override
public boolean onContextItemSelected(@NonNull MenuItem item) {
    AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo) item.getMenuInfo();
    switch (item.getItemId()){
        case R.id.delete_item:
            nombres.remove(info.position);
            adapter.notifyDataSetChanged();
            return true;
        default:
            return super.onContextItemSelected(item);
    }
}
```

Context Menu

Y además, tenemos que añadir en el método onCreate(), una llamada al método **registerForContextMenu(view)**.

Java

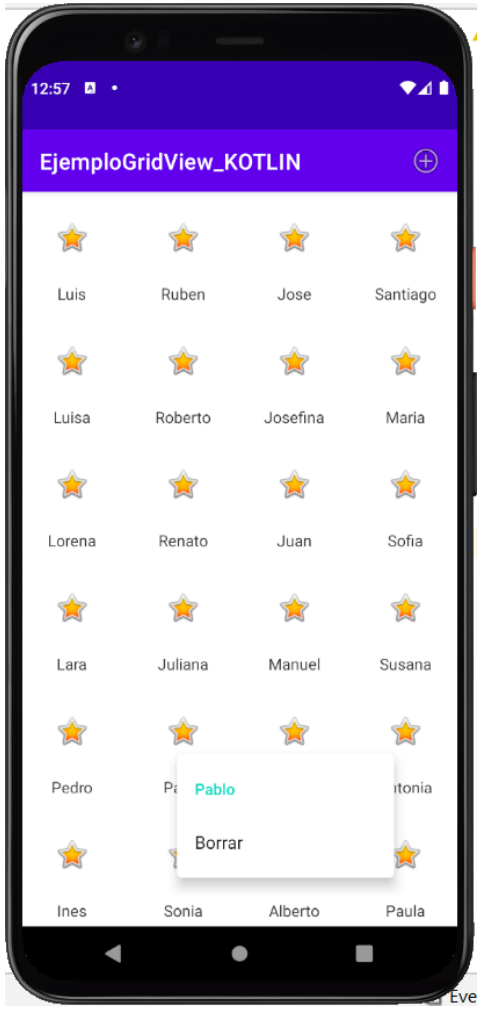
```
    adapter = new MyAdapter( context: th
    gridView.setAdapter(adapter);
    registerForContextMenu(gridView);
}
```

Kotlin

```
    }.show()
})
registerForContextMenu(gridView)
```

Programación Android

Context Menu



android

Ejercicio

Programación Android

Ejercicio

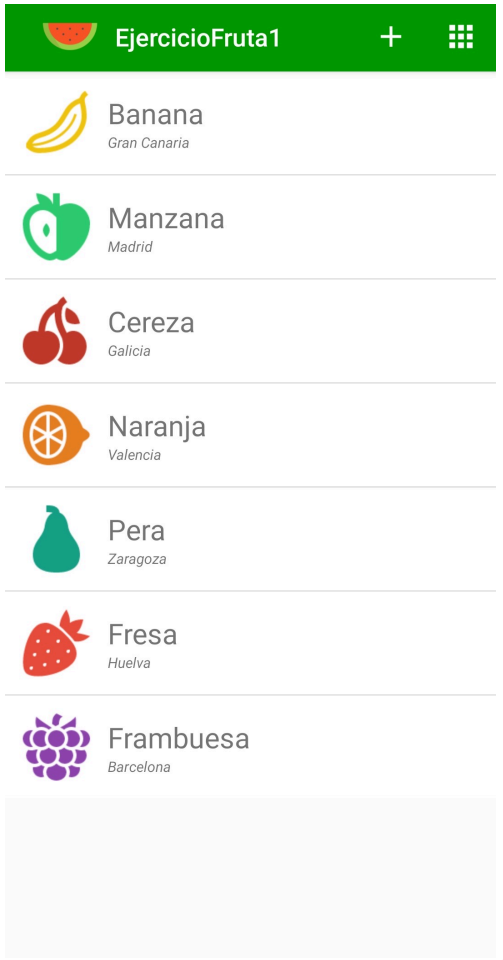
Crear una app en la que tendremos un **ListView,** por defecto al abrir la app, que contendrá frutas.

Fruta, será un modelo POJO que nosotros crearemos, con 3 atributos, nombre, origen e icono(es un **Int**, se le pasará una referencia con **R.id** a un icono que previamente tenemos que crear con Android Studio).

Crear **3 paquetes**, con los siguientes nombres: **activities**, **adapters**, **models**. Alojar en cada uno el contenido apropiado.

Tendremos **3 opciones** en el menú, aunque el usuario solo verá **2 a la vez** (opción añadir y un button):

- **Añadir fruta** (siempre mismo icono, mismo origen, y mismo nombre cambiando el número al final, con un contador),
- **GridView button** o **ListView button**, ListView button cuando el GridView esté renderizado, y GridView button cuando el ListView esté renderizado.

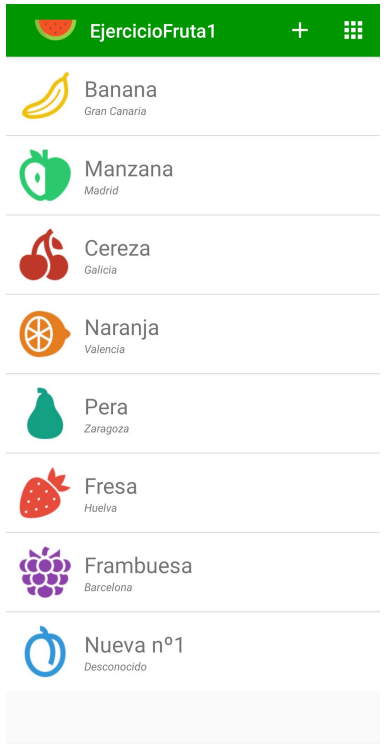


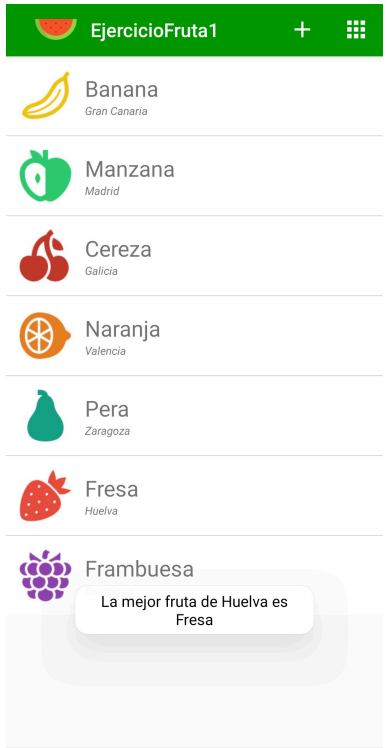
android

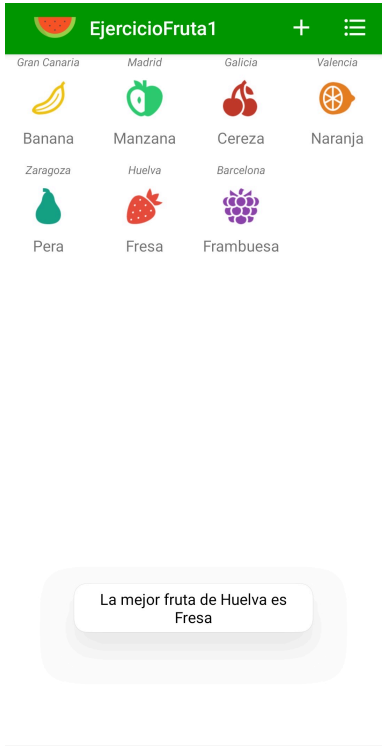
Programación Android

Ejercicio

Añadimos la función **click** sobre frutas en ambos views, aunque crearemos sólo un método y será aplicado para ambos. Aplica la interfaz “AdapterView.OnItemClickListener”. Enseñamos un **mensaje** para las frutas precargadas, pero para las nuevas añadidas, otro mensaje.





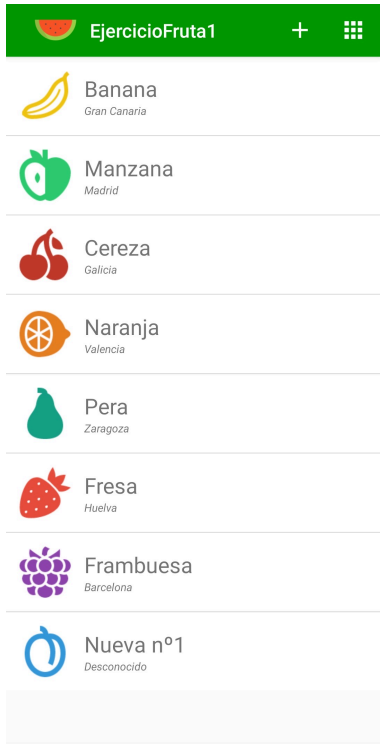


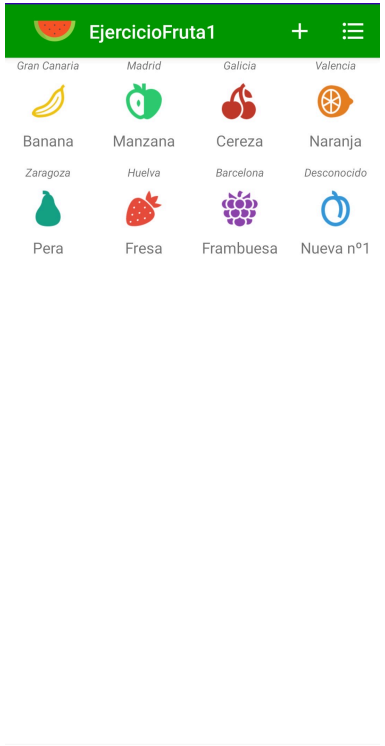
android

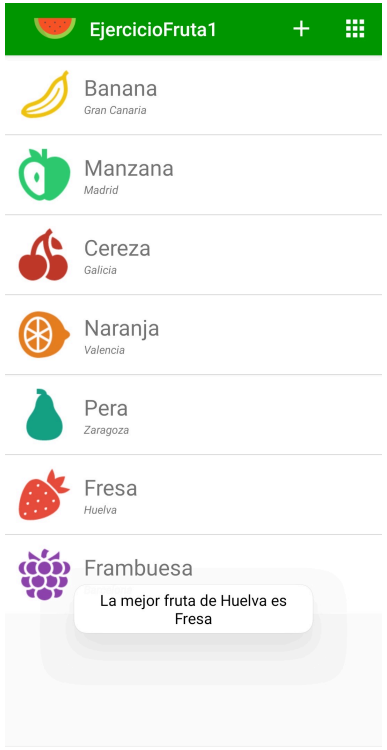
Programación Android

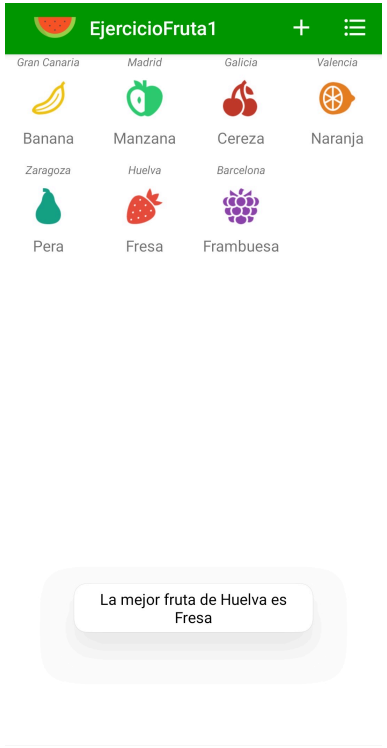
Ejercicio

Crea un **método** para hacer el switch entre List y Grid View. La lógica del método debe seguir lo siguiente: Se le pasa como parámetro un valor que indica a qué View queremos cambiar, entonces comprobaremos si ese View no está visible, para proseguir.







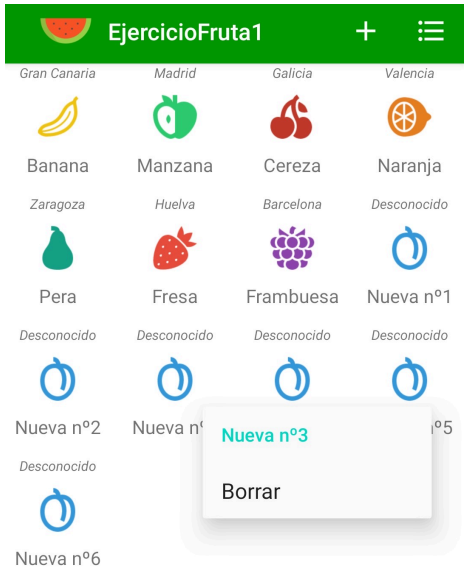


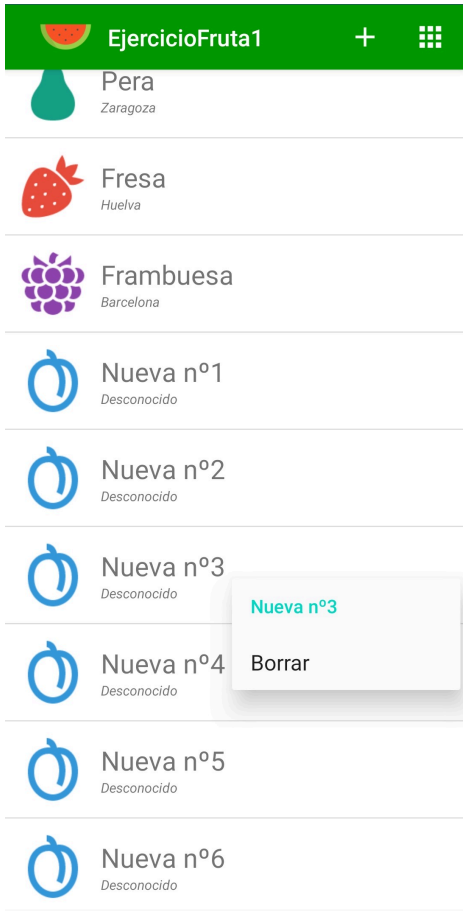
android

Programación Android

Ejercicio

Por último, añadiremos un **Context Menu** para borrar elementos. Este Context Menu debe tener como título el nombre de la fruta pulsada.





android