

**MP_0489 Programación
multimedia y dispositivos móviles
UF2. Programación de aplicaciones
para dispositivos móviles**

2.8. Proyecto de aplicación

Índice

☰	Objetivos	3
☰	Crear el proyecto	4
☰	Almacenar datos	10
☰	Recuperar datos	14
☰	Borrar tareas	19
☰	Resumen	24

Objetivos

Con esta unidad perseguimos los siguientes objetivos:

1

Generar el proyecto de una aplicación con una lista de tareas.

2

Trabajar con bases de datos.

¡Ánimo y adelante!

Crear el proyecto

A lo largo de esta unidad vamos a crear una **aplicación para gestionar una lista de tareas**.

Crear el nuevo proyecto

1

Para crear un nuevo proyecto, abrimos Android Studio y hacemos clic en *Iniciar un nuevo proyecto de Android Studio*. Nombramos la aplicación “TodoList”.

2

Seleccionamos *Actividad vacía* y mantenemos el nombre como *MainActivity*.

3

Cuando Android Studio termine de generar el proyecto, tendremos la aplicación predeterminada “**Hola, Mundo!**”.

Modificar el *layout*

Vamos a modificar el archivo del *layout* para agregar un *ListView*, que contendrá un elemento de tareas pendientes en cada fila.

1

Para hacer esto, **reemplazamos el elemento *TextView*** con el siguiente código:

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

Ahora, definimos un elemento de lista, que representará una tarea en la interfaz de usuario.

2

Creamos un nuevo archivo de diseño en la carpeta */res/layout* llamado *item_todo.xml*.

3

Agregamos dos elementos a este archivo, un *TextView* para mostrar la tarea y un *Button* “**Listo**” para eliminar la tarea.

4

Agregamos también este código a *item_todo.xml*, reemplazando el que tengamos actualmente:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.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"
    android:layout_gravity="center_vertical">

    <TextView
        android:id="@+id/task_title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:text="Hello"
        android:textSize="20sp" />

    <Button
        android:id="@+id/task_delete"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentRight="true"
        android:text="Done"
        app:layout_constraintEnd_toEndOf="parent" />

</android.support.constraint.ConstraintLayout>

```

Además, la aplicación necesita un **elemento de menú** para permitir que el usuario agregue más tareas.

5

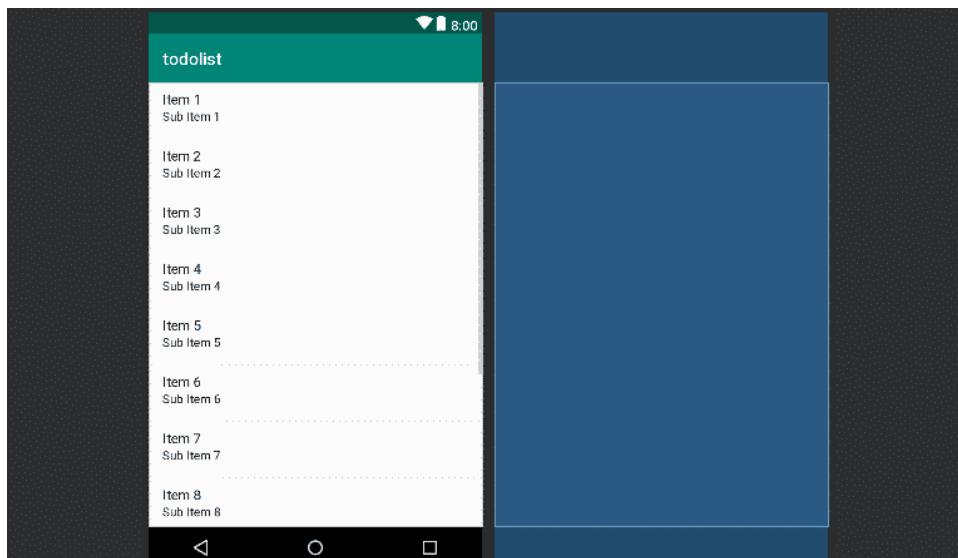
Agregamos un archivo `main_menu.xml` en el directorio `/res/menu` con el siguiente código:

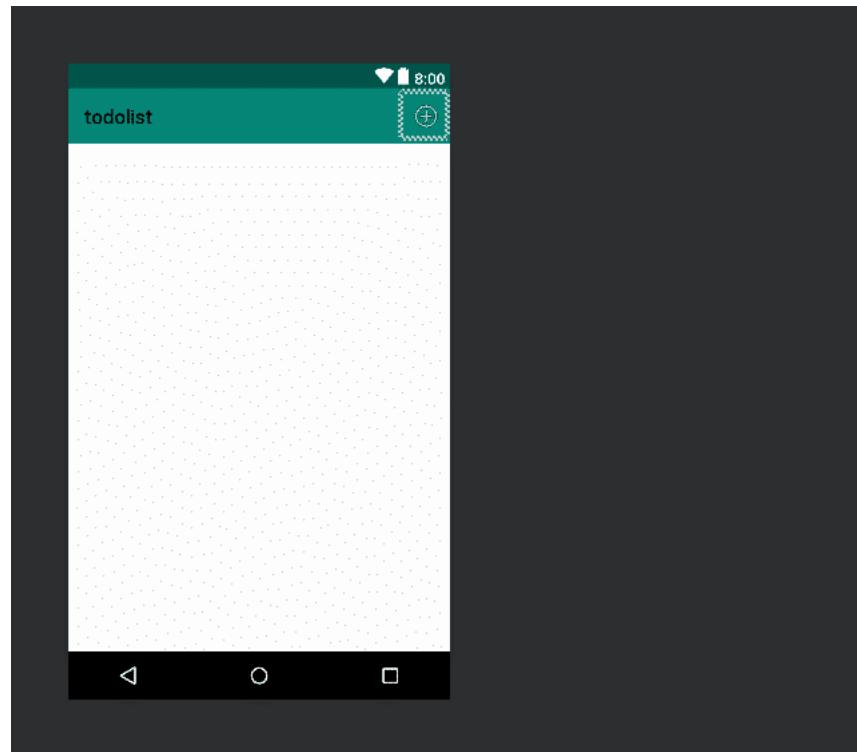
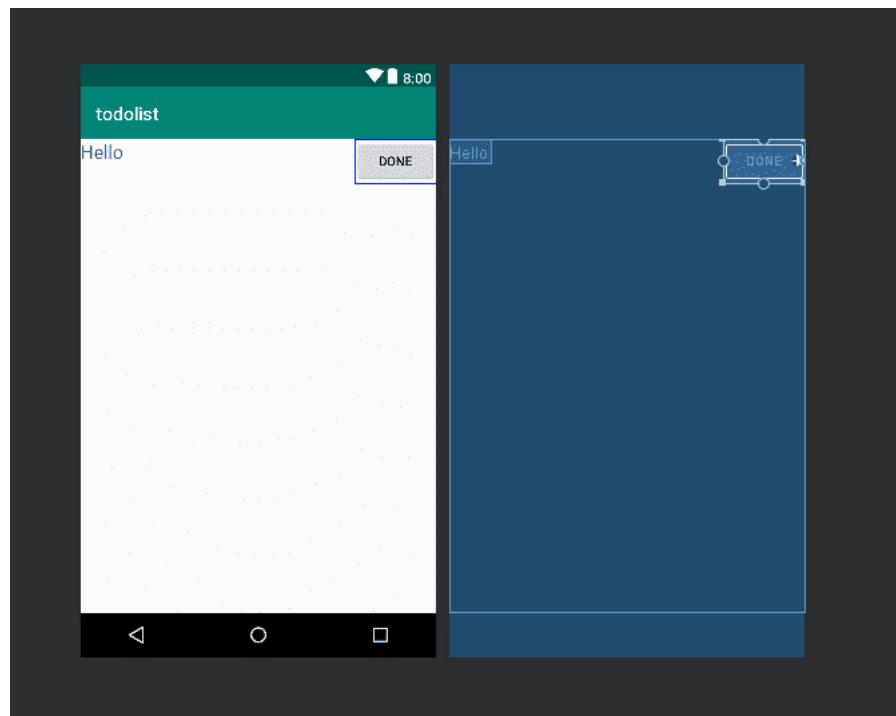
```

<?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/action_add_task"
        android:icon="@android:drawable/ic_menu_add"
        android:title="Add Task"
        app:showAsAction="always" />
</menu>

```

El resultado en la vista de diseño debería ser similar a este:





Implementar otras funcionalidades

1

Vamos a crear un método para mostrar el menú en la actividad principal, `onCreateOptionsMenu()`.

```
package com.miejemplo.todolist;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;

public class MainActivity extends AppCompatActivity {

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

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

}
```

2

También crearemos el método `onOptionsItemSelected()` para reaccionar a diferentes interacciones de los usuarios con los elementos del menú.

```
package com.miejemplo.todolist;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;

public class MainActivity extends AppCompatActivity {

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

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_add_task:
                return true;

            default:
                return super.onOptionsItemSelected(item);
        }
    }
}
```

3

A continuación, añadiremos un *AlertDialog* para obtener la tarea del usuario cuando se haga clic en el botón *Agregar elemento*. Lo haremos dentro del método *onOptionsItemSelected()*:

```
package com.miejemplo.todolist;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.content.DialogInterface;
import android.widget.EditText;
import android.support.v7.app.AlertDialog;

public class MainActivity extends AppCompatActivity {

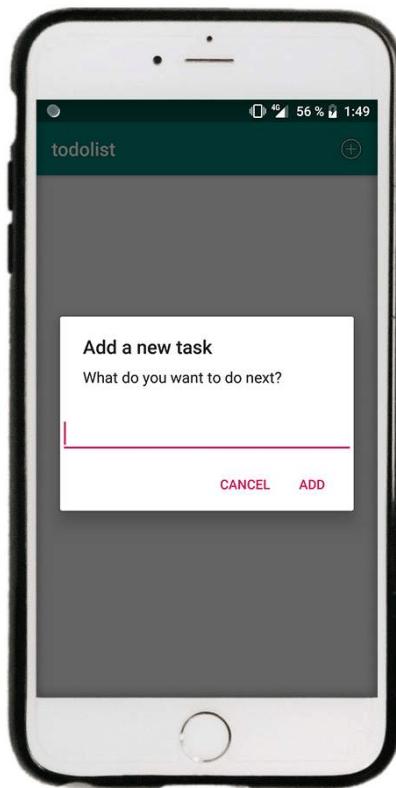
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_add_task:
                final EditText taskEditText = new EditText(this);
                AlertDialog dialog = new AlertDialog.Builder(this)
                    .setTitle("Add a new task")
                    .setMessage("What do you want to do next?")
                    .setView(taskEditText)
                    .setPositiveButton("Add", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            String task = String.valueOf(taskEditText.getText());
                        }
                    })
                    .setNegativeButton("Cancel", null)
                    .create();
                dialog.show();

                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }
}
```

Este es el resultado si ejecutamos nuestra aplicación:



En el siguiente apartado te mostraremos los pasos a seguir para **almacenar las tareas**.

Almacenar datos

Continuamos con el desarrollo de nuestra aplicación aprendiendo a realizar el **almacenamiento de datos**.

Almacenamiento de datos

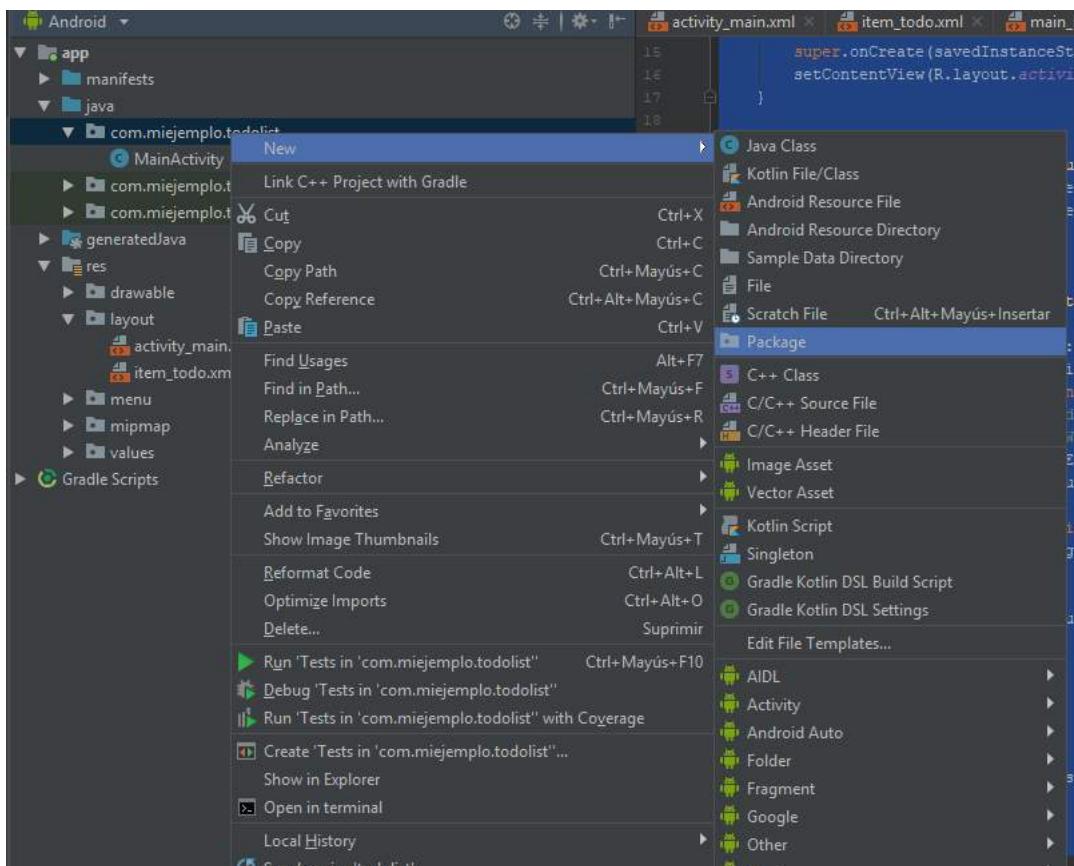
Como sabemos, Android viene con una base de datos SQLite incrustada. Esta base de datos necesita una tabla antes de poder almacenar cualquier tarea, a la que vamos a llamar "TaskTable".

1

Creamos una nueva carpeta db, como vemos en la imagen, en la misma ubicación que MainActivity.java.

2

Luego, creamos una nueva clase *TaskContract*, con el nombre de archivo TaskContract.java dentro de ella



3

Agregamos este código a TaskContract.java:

```
package com.miejemplo.todolist.db;

import android.provider.BaseColumns;

public class TaskContract {
    public static final String DB_NAME = "com.miejemplo.todolist.db";
    public static final int DB_VERSION = 1;

    public class TaskEntry implements BaseColumns {
        public static final String TABLE = "tasks";

        public static final String COL_TASK_TITLE = "title";
    }
}
```

La clase *TaskContract* define constantes que se utilizan para acceder a los datos en la base de datos.

4

También necesitaremos una clase auxiliar llamada *TaskDbHelper* para abrir la base de datos. Crearemos esta clase en la carpeta db, agregando el siguiente código:

```
package com.miejemplo.todolist.db;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class TaskDbHelper extends SQLiteOpenHelper {

    public TaskDbHelper(Context context) {
        super(context, TaskContract.DB_NAME, null, TaskContract.DB_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        String createTable = "CREATE TABLE " + TaskContract.TaskEntry.TABLE + " (" +
            TaskContract.TaskEntry._ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
            TaskContract.TaskEntry.COL_TASK_TITLE + " TEXT NOT NULL);";

        db.execSQL(createTable);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TaskContract.TaskEntry.TABLE);
        onCreate(db);
    }
}
```

Observa que entre las líneas 15 y 17 estamos ejecutando esta consulta SQL:

```
CREATE TABLE tasks (
    _id INTEGER PRIMARY KEY AUTOINCREMENT,
    title TEXT NOT NULL
);
```

Ahora necesitamos adaptar `MainActivity.java` para almacenar los datos en la base de datos.

5

Añadimos este código en `DialogInterface.OnClickListener()` para el botón Agregar de `AlertDialog`:

```
package com.miejemplo.todolist;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.content.DialogInterface;
import android.widget.EditText;
import android.support.v7.app.AlertDialog;
import android.database.sqlite.SQLiteDatabase;
import android.content.ContentValues;

import com.miejemplo.todolist.db.TaskContract;
import com.miejemplo.todolist.db.TaskDbHelper;

public class MainActivity extends AppCompatActivity {
    private TaskDbHelper mHelper;

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

        mHelper = new TaskDbHelper(this);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_add_task:
                final EditText taskEditText = new EditText(this);
                AlertDialog dialog = new AlertDialog.Builder(this)
                    .setTitle("Add a new task")
                    .setMessage("What do you want to do next?")
                    .setView(taskEditText)
                    .setPositiveButton("Add", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            String task = String.valueOf(taskEditText.getText());
                            SQLiteDatabase db = mHelper.getWritableDatabase();
                            ContentValues values = new ContentValues();
                            values.put(TaskContract.TaskEntry.COL_TASK_TITLE, task);
                            db.insertWithOnConflict(TaskContract.TaskEntry.TABLE,
                                null,
                                values,
                                SQLiteDatabase.CONFLICT_REPLACE);
                            db.close();
                        }
                    })
                    .setNegativeButton("Cancel", null)
                    .create();
                dialog.show();

                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }
}
```

Hemos agregado una instancia privada de `TaskDbHelper` en la clase `MainActivity`:

```
private TaskDbHelper mHelper;
```

Y la hemos inicializado en el método `onCreate()`:

```
mHelper = new TaskDbHelper(this);
```

Recuperar datos

La **recuperación de datos** también es fundamental en el desarrollo de nuestra aplicación.

Recuperación de datos

Vamos a recuperar todos los datos de la base de datos y a mostrarlos en la vista principal.

1

Para ello, crearemos un nuevo método, que se llamará *updateUI()*.

```
package com.miejemplo.todolist;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.content.DialogInterface;
import android.widget.EditText;
import android.support.v7.app.AlertDialog;
import android.database.sqlite.SQLiteDatabase;
import android.content.ContentValues;
import android.widget.ArrayAdapter;
import android.database.Cursor;
import android.widget.ListView;

import java.util.ArrayList;

import com.miejemplo.todolist.db.TaskContract;
import com.miejemplo.todolist.db.TaskDbHelper;

public class MainActivity extends AppCompatActivity {
    private TaskDbHelper mHelper;
    private ListView mTaskListView;
    private ArrayAdapter<String> mAdapter;

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

        mHelper = new TaskDbHelper(this);
        mTaskListView = (ListView) findViewById(R.id.list_todo);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }
}
```

```

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.action_add_task:
            final EditText taskEditText = new EditText(this);
            AlertDialog dialog = new AlertDialog.Builder(this)
                .setTitle("Add a new task")
                .setMessage("What do you want to do next?")
                .setView(taskEditText)
                .setPositiveButton("Add", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        String task = String.valueOf(taskEditText.getText());
                        SQLiteOpenHelper db = mHelper.getWritableDatabase();
                        ContentValues values = new ContentValues();
                        values.put(TaskContract.TaskEntry.COL_TASK_TITLE, task);
                        db.insertWithOnConflict(TaskContract.TaskEntry.TABLE,
                                null,
                                values,
                                SQLiteDatabase.CONFLICT_REPLACE);
                        db.close();
                    }
                })
                .setNegativeButton("Cancel", null)
                .create();
            dialog.show();

            return true;

        default:
            return super.onOptionsItemSelected(item);
    }
}

private void updateUI() {
    ArrayList<String> taskList = new ArrayList<>();
    SQLiteOpenHelper db = mHelper.getReadableDatabase();
    Cursor cursor = db.query(TaskContract.TaskEntry.TABLE,
            new String[]{TaskContract.TaskEntry._ID, TaskContract.TaskEntry.COL_TASK_TITLE},
            null, null, null, null, null);
    while (cursor.moveToNext()) {
        int idx = cursor.getColumnIndex(TaskContract.TaskEntry.COL_TASK_TITLE);
        taskList.add(cursor.getString(idx));
    }

    if (mAdapter == null) {
        mAdapter = new ArrayAdapter<>(this,
                R.layout.item_todo,
                R.id.task_title,
                taskList);
        mTaskListView.setAdapter(mAdapter);
    } else {
        mAdapter.clear();
        mAdapter.addAll(taskList);
        mAdapter.notifyDataSetChanged();
    }

    cursor.close();
    db.close();
}
}

```

Hemos creado una referencia, agregando una instancia privada de *ListView* en la clase *MainActivity*:

```
private ListView mTaskListView;
```

Y hemos inicializado esa referencia, agregando esta línea de código al método `onCreate()`, justo después de `mHelper`:

```
mTaskListView = (ListView) findViewById(R.id.list_todo);
```

Además, hemos agregado este campo privado a la clase `MainActivity`:

```
private ArrayAdapter<String> mAdapter;
```

Este `ArrayAdapter` ayudará a llenar el `ListView` con los datos. Lo que hacemos con el método `updateUI()` es agregarlos a un `ArrayList` de `Strings`.

2

Ahora, comprobamos si `mAdapter` está creado o no.

- Si no lo está, y `mAdapter` es `null`, lo creamos como adaptador de `ListView`.
- Si el adaptador ya está creado (lo que implica que está asignado a `ListView`), lo vaciamos, lo rellenamos nuevamente y notificamos a la vista que los datos han cambiado. La vista se volverá a pintar en la pantalla con los nuevos datos.

3

Para ver los datos actualizados, debemos llamar al método `updateUI()` cada vez que los datos cambien. Lo haremos en dos lugares: `onCreate()` y `AlertDialog`.

```
package com.miejemplo.todolist;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.content.DialogInterface;
import android.widget.EditText;
import android.support.v7.app.AlertDialog;
import android.database.sqlite.SQLiteDatabase;
import android.content.ContentValues;
import android.widget.ArrayAdapter;
import android.database.Cursor;
import android.widget.ListView;

import java.util.ArrayList;

import com.miejemplo.todolist.db.TaskContract;
import com.miejemplo.todolist.db.TaskDbHelper;

public class MainActivity extends AppCompatActivity {
```

```

private TaskDbHelper mHelper;
private ListView mTaskListView;
private ArrayAdapter<String> mAdapter;

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

    mHelper = new TaskDbHelper(this);
    mTaskListView = (ListView) findViewById(R.id.list_todo);

    updateUI();
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.main_menu, menu);
    return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.action_add_task:
            final EditText taskEditText = new EditText(this);
            AlertDialog dialog = new AlertDialog.Builder(this)
                .setTitle("Add a new task")
                .setMessage("What do you want to do next?")
                .setView(taskEditText)
                .setPositiveButton("Add", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        String task = String.valueOf(taskEditText.getText());
                        SQLiteDatabase db = mHelper.getWritableDatabase();
                        ContentValues values = new ContentValues();
                        values.put(TaskContract.TaskEntry.COL_TASK_TITLE, task);
                        db.insertWithOnConflict(TaskContract.TaskEntry.TABLE,
                                null,
                                values,
                                SQLiteDatabase.CONFLICT_REPLACE);
                        db.close();
                        updateUI();
                    }
                })
                .setNegativeButton("Cancel", null)
                .create();
            dialog.show();

            return true;

        default:
            return super.onOptionsItemSelected(item);
    }
}

private void updateUI() {
    ArrayList<String> taskList = new ArrayList<>();
    SQLiteDatabase db = mHelper.getReadableDatabase();
    Cursor cursor = db.query(TaskContract.TaskEntry.TABLE,
            new String[]{TaskContract.TaskEntry._ID, TaskContract.TaskEntry.COL_TASK_TITLE},
            null, null, null, null);
    while (cursor.moveToNext()) {
        int idx = cursor.getColumnIndex(TaskContract.TaskEntry.COL_TASK_TITLE);
        taskList.add(cursor.getString(idx));
    }

    if (mAdapter == null) {
        mAdapter = new ArrayAdapter<>(this,
                R.layout.item_todo,
                R.id.task_title,
                taskList);
    }
}

```

```
mTaskListView.setAdapter(mAdapter);
} else {
    mAdapter.clear();
    mAdapter.addAll(taskList);
    mAdapter.notifyDataSetChanged();
}

cursor.close();
db.close();
}

}
```

Borrar tareas

Hemos creado una aplicación para gestionar una lista de tareas.

Pero, recuerda que una vez realizada cada tarea, **debe ser eliminada de la lista**.

Eliminar tareas

1

Abrimos el *layout item_todo.xml* y modificamos *button* para que tenga este aspecto:

```
<Button  
    android:id="@+id/task_delete"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentEnd="true"  
    android:layout_alignParentRight="true"  
    android:text="Done"  
    app:layout_constraintEnd_toEndOf="parent"  
    android:onClick="deleteTask" />
```

2

Cuando hagamos clic en el botón, llamaremos al método *deleteTask()* en la clase *MainActivity*.

```
package com.miejemplo.todolist;  
  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.Menu;  
import android.view.MenuItem;  
import android.content.DialogInterface;  
import android.widget.EditText;  
import android.support.v7.app.AlertDialog;  
import android.database.sqlite.SQLiteDatabase;  
import android.content.ContentValues;  
import android.widget.ArrayAdapter;  
import android.database.Cursor;  
import android.widget.ListView;  
import android.view.View;  
import android.widget.TextView;  
  
import java.util.ArrayList;
```

```

import com.miejemplo.todolist.db.TaskContract;
import com.miejemplo.todolist.db.TaskDbHelper;

public class MainActivity extends AppCompatActivity {
    private TaskDbHelper mHelper;
    private ListView mTaskListView;
    private ArrayAdapter<String> mAdapter;

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

        mHelper = new TaskDbHelper(this);
        mTaskListView = (ListView) findViewById(R.id.list_todo);

        updateUI();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_add_task:
                final EditText taskEditText = new EditText(this);
                AlertDialog dialog = new AlertDialog.Builder(this)
                    .setTitle("Add a new task")
                    .setMessage("What do you want to do next?")
                    .setView(taskEditText)
                    .setPositiveButton("Add", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            String task = String.valueOf(taskEditText.getText());
                            SQLiteDatabase db = mHelper.getWritableDatabase();
                            ContentValues values = new ContentValues();
                            values.put(TaskContract.TaskEntry.COL_TASK_TITLE, task);
                            db.insertWithOnConflict(TaskContract.TaskEntry.TABLE,
                                null,
                                values,
                                SQLiteDatabase.CONFLICT_REPLACE);
                            db.close();
                            updateUI();
                        }
                    })
                    .setNegativeButton("Cancel", null)
                    .create();
                dialog.show();

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

    public void deleteTask(View view) {
        View parent = (View) view.getParent();
        TextView taskTextView = (TextView) parent.findViewById(R.id.task_title);
        String task = String.valueOf(taskTextView.getText());
        SQLiteDatabase db = mHelper.getWritableDatabase();
        db.delete(TaskContract.TaskEntry.TABLE,
            TaskContract.TaskEntry.COL_TASK_TITLE + " = ?",
            new String[]{task});
        db.close();
        updateUI();
    }
}

```

```

private void updateUI() {
    ArrayList<String> taskList = new ArrayList<>();
    SQLiteDatabase db = mHelper.getReadableDatabase();
    Cursor cursor = db.query(TaskContract.TaskEntry.TABLE,
        new String[]{TaskContract.TaskEntry._ID, TaskContract.TaskEntry.COL_TASK_TITLE},
        null, null, null, null, null);
    while (cursor.moveToFirst()) {
        int idx = cursor.getColumnIndex(TaskContract.TaskEntry.COL_TASK_TITLE);
        taskList.add(cursor.getString(idx));
    }

    if (mAdapter == null) {
        mAdapter = new ArrayAdapter<>(this,
            R.layout.item_todo,
            R.id.task_title,
            taskList);
        mTaskListView.setAdapter(mAdapter);
    } else {
        mAdapter.clear();
        mAdapter.addAll(taskList);
        mAdapter.notifyDataSetChanged();
    }

    cursor.close();
    db.close();
}
}

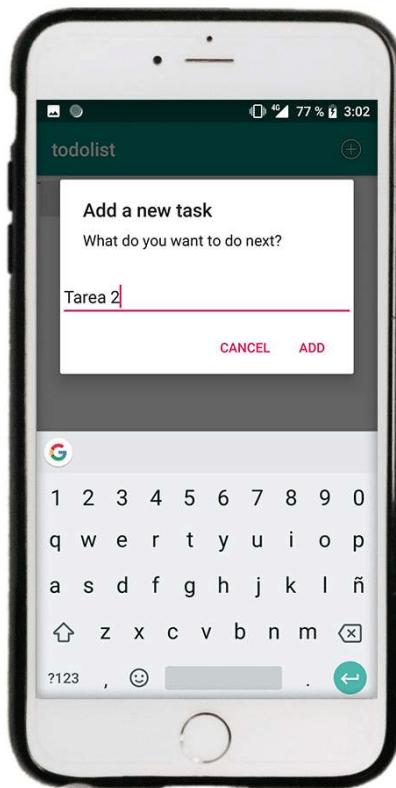
```

3

Al hacer clic en el botón **Done** se eliminará la tarea de la lista y de la base de datos SQLite.

Este es el aspecto de las distintas pantallas de nuestra aplicación finalizada.







Resumen

Hemos terminado la lección, repasemos los puntos más importantes que hemos tratado.

- A lo largo de esta unidad hemos aprendido a crear una aplicación sencilla con una lista de tareas.
- Hemos visto cómo trabajar con cuadros de diálogo en la interfaz de usuario.
- Hemos utilizado SQLite para almacenar los datos.
- Y, para finalizar, hemos visualizado los datos guardados y aprendido cómo eliminarlos.

