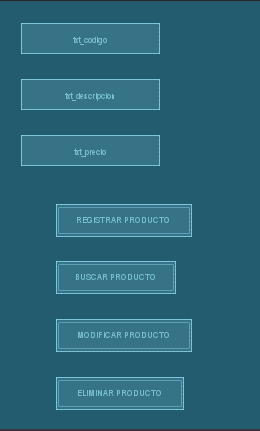
En esta clase se vio como crear una aplicación con un motor de bases de datos que necesita poca configuración llamado SQLite.

Cada vez que queramos usar una base de datos de esta forma tendremos que crear una nueva java class SQLiteOpenHelper.

Y se crean los dos métodos onCreate y onUpgrade.

package com.example.base\_datos\_1;  
import android.content.Context;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
  
import androidx.annotation.Nullable;  
  
public class AdminSQLiteOpenHelper extends SQLiteOpenHelper{  
 public AdminSQLiteOpenHelper(@Nullable Context context, @Nullable String name, @Nullable SQLiteDatabase.CursorFactory factory, int version) {  
 super(context, name, factory, version);  
 }  
 @Override  
 public void onCreate(SQLiteDatabase BaseDeDatos) {  
  
 BaseDeDatos.execSQL("create table articulos(codigo int primary key, descripcion text, precio real)");  
 }  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
  
 }  
}

<resources>  
 <string name="app\_name">Base\_Datos\_1</string>  
 <string name="txt\_codigo">cod producto</string>  
 <string name="txt\_descripcion">descripcion producto</string>  
 <string name="txt\_precio">precio prod</string>  
 <string name="botonRegistrar">Registrar Producto</string>  
 <string name="botonBuscar">Buscar Producto</string>  
 <string name="botonEliminar">Eliminar Producto</string>  
 <string name="botonModificar">Modificar Producto</string>  
</resources>



Después en la clase main ponemos el método para registrar los productos

public void Registrar(View view){  
 AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper(this, "administracion", null, 1);  
 SQLiteDatabase BaseDeDatos = admin.getWritableDatabase();

Y para la validación del campo se hace con los if correspondientes.

if(!codigo.isEmpty() && !descripcion.isEmpty() && !precio.isEmpty()){  
 ContentValues registro = new ContentValues();  
  
 registro.put("codigo", codigo);  
 registro.put("descripcion", descripcion);  
 registro.put("precio", precio);  
  
 BaseDeDatos.insert("articulos",null, registro);  
  
 BaseDeDatos.close();  
 et\_codigo.setText("");  
 et\_descripcion.setText("");  
 et\_precio.setText("");  
  
 Toast.*makeText*(this, "Registro Exitoso", Toast.*LENGTH\_LONG*).show();  
 } else{  
 Toast.*makeText*(this, "Debes llenar los campos", Toast.*LENGTH\_LONG*).show();  
 }  
}

Para el método buscar se crea el método y se crea el mismo objeto que antes SQLiteOpenHelper.

public void Buscar(View view){  
  
 AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper(this, "administracion", null, 1);  
 SQLiteDatabase BaseDeDatabase = admin.getWritableDatabase();

Y para poder buscar tenemos que recuperar el valor que habíamos usado con anterioridad, en el código.

Y esto hay que restringirlo con un if para que se introduzcan los campos correspondientes.

String codigo = et\_codigo.getText().toString();  
  
if(!codigo.isEmpty()){  
 Cursor fila = BaseDeDatabase.rawQuery  
 ("select descripcion, precio from articulos where codigo =" + codigo, null);  
 if(fila.moveToFirst()){  
 et\_descripcion.setText(fila.getString(0));  
 et\_precio.setText(fila.getString(1));  
 BaseDeDatabase.close();  
  
 Toast.*makeText*(this, "El articulo no existe", Toast.*LENGTH\_SHORT*).show();  
 BaseDeDatabase.close();  
 }  
} else{  
 Toast.*makeText*(this, "Debes introducir el codigo de articulo", Toast.*LENGTH\_LONG*).show();  
}

