|  |  |
| --- | --- |
|  | **UNIVERSIDAD DON BOSCO**  **FACULTAD DE INGENIERÍA – ESCUELA DE COMPUTACIÓN** |
|  | TALLER ANDROID  **TEMA V – Base de Datos**  Uso de Room y validación de formularios |

**I Objetivos**

* Utilizar Room como herramienta de persistencia de base de datos.
* Incluir saripaar para realizar validaciones de campos de entrada.

**II Introducción**

**Room: a SQL object mapping library**

Room provee una capa de abstración sobre SQLite y permite gestionar las bases de datos bajo el modelo ORM (Object-Relational mapping). Esto implica que el desarrollador no tendrá que utilizar la configuración y sintaxis de SQLite, más bien, utilizará la librería de Room para realizar las operaciones típicas sobre una base de datos (CRUD). Room ha pasado a formar parte de las librerías recomendadas para los nuevos desarrollos en Android.

**III Desarrollo**

1. Agregar soporte de Room, design y saripaar:

|  |
| --- |
| */\*Soporte base de datos con Room\*/* implementation **"android.arch.persistence.room:runtime:1.0.0"** annotationProcessor **"android.arch.persistence.room:compiler:1.0.0"**  */\*Librería de Design (boton flotante)\*/* implementation **'com.android.support:design:28.0.0'**  */\*Librería de validaciones\*/* implementation **'com.mobsandgeeks:android-saripaar:2.0.3'** |

1. Modificar el POJO RecyclerCenter:



1. **Agregar una nueva actividad denominada AddRecyclingCenterActivity**
2. Vamos a agregar una interface que nos permitirá realizar las operaciones de la base de datos. Room utiliza la interface para realizar las operaciones en backgroud.

RecyclingCenterDAO.java

|  |
| --- |
| **package** sv.edu.udb.reciclaje.data;  **import** android.arch.persistence.room.Dao; **import** android.arch.persistence.room.Delete; **import** android.arch.persistence.room.Insert; **import** android.arch.persistence.room.Query; **import** android.arch.persistence.room.Update;  **import** java.util.List;  @Dao **public interface** RecyclingCenterDAO {   @Insert  **public void** insert(RecyclingCenter recyclingCenter);   @Update  **public void** update(RecyclingCenter recyclingCenter);   @Delete  **public void** delete(RecyclingCenter recyclingCenter);   @Query(**"SELECT \* FROM recycler\_center"**)  **public** List<RecyclingCenter> getRecyclingCenterList();   @Query(**"SELECT \* FROM recycler\_center where id = :id"**)  **public** RecyclingCenter getRecyclingCenter(Long id);  } |

1. Crearemos una clase abstracta AppDatabase.java para inicialiar el DAO.

AppDatabase.java

|  |
| --- |
| **package** sv.edu.udb.reciclaje.data;  **import** android.arch.persistence.room.Database; **import** android.arch.persistence.room.RoomDatabase;  @Database(entities = {RecyclingCenter.**class**},version = 1) **public abstract class** AppDatabase **extends** RoomDatabase {  **public abstract** RecyclingCenterDAO getRecyclingCenterDAO(); } |

Con esto será suficiente, ya podemos realizar operaciones sobre la base de datos local que hemos creado.

1. Nuestro viejo RecyclingCenterActivity.java ha sido modificado. Ahora contiene el siguiente código:

Nota: Se ha subrayado en amarillo los cambios principales.

|  |
| --- |
| **package** sv.edu.udb.reciclaje;  **import** android.arch.persistence.room.Room; **import** android.content.Intent; **import** android.support.v7.app.ActionBar; **import** android.support.v7.app.AppCompatActivity; **import** android.os.Bundle; **import** android.support.v7.widget.LinearLayoutManager; **import** android.support.v7.widget.RecyclerView; **import** android.view.View;  **import** sv.edu.udb.reciclaje.data.AppDatabase; **import** sv.edu.udb.reciclaje.data.RecyclingCenter; **import** sv.edu.udb.reciclaje.data.RecyclingCenterDAO; **import** sv.edu.udb.reciclaje.data.RecyclingData; **import** sv.edu.udb.reciclaje.util.RecyclingCenterAdapter;  **public class** RecyclingCenterActivity **extends** AppCompatActivity{   RecyclerView **recyclerView**;   *//Dejamos de utilizarlo  //RecyclingData recyclingData = new RecyclingData();* @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);    setContentView(R.layout.***activity\_recycling\_center***);   **recyclerView** = findViewById(R.id.***recyclerview***);   **recyclerView**.setLayoutManager(**new** LinearLayoutManager(**this**));   *//Modificamos el antiguo contenido  //RecyclingCenterAdapter adapter = new RecyclingCenterAdapter(recyclingData.getRecyclingCenterList(), this);* }    @Override  **protected void** onResume() {  **super**.onResume();   AppDatabase database = Room.*databaseBuilder*(**this**, AppDatabase.**class**, **"mydb"**)  .allowMainThreadQueries()  .build();   RecyclingCenterDAO recyclingCenterDAO = database.getRecyclingCenterDAO();  RecyclingCenterAdapter adapter = **new** RecyclingCenterAdapter(recyclingCenterDAO.getRecyclingCenterList(), **this**);   **recyclerView**.setAdapter(adapter);   }   **public void** add(View view) {  Intent intent = **new** Intent(**this**, AddRecyclingCenterActivity.**class**);  startActivity(intent);  } } |

1. activty\_recycling\_center.xml contiene ahora un floating button:

|  |  |
| --- | --- |
|  | *<?***xml version="1.0" encoding="utf-8"***?>* <**RelativeLayout 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=".RecyclingCenterActivity"** >   <**android.support.v7.widget.RecyclerView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/recyclerview"** >   </**android.support.v7.widget.RecyclerView**>   <**android.support.design.widget.FloatingActionButton  android:id="@+id/fab"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_gravity="bottom|end"  android:layout\_margin="20dp"  android:layout\_alignParentBottom="true"  android:layout\_alignParentRight="true"  android:src="@android:drawable/ic\_menu\_add"  android:onClick="add"** />   </**RelativeLayout**> |

1. Editaremos la actividad de agregar un nuevo sitio.

activity\_add\_recycling\_center.xml

|  |  |
| --- | --- |
|  | *<?***xml version="1.0" encoding="utf-8"***?>* <**android.support.v7.widget.CardView 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=".AddRecyclingCenterActivity"  app:cardElevation="10dp"  app:cardCornerRadius="10dp"  android:layout\_margin="5dp"  app:contentPadding="5dp"** > <**LinearLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  tools:context=".AddRecyclingCenterActivity"  android:orientation="vertical"** >   <**EditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/name"  android:hint="Nombre"** />  <**EditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/address"  android:hint="Direccion"** />  <**EditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/imagesiteurl"  android:hint="URL Imagen"** />  <**EditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/email"  android:hint="email"** />   <**EditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/latitude"  android:hint="latitud"  android:inputType="numberDecimal"** />   <**EditText  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:id="@+id/longitude"  android:hint="longitud"  android:inputType="numberDecimal"** />   <**Button  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:onClick="add"  android:background="#C20E0E"  android:text="Agregar"  android:textColor="#fff"  android:layout\_margin="10dp"** />   </**LinearLayout**>  </**android.support.v7.widget.CardView**> |

1. Editaremos la clase de agregar AddRecyclingCenterActivity.java:

|  |
| --- |
| **package** sv.edu.udb.reciclaje;  **import** android.arch.persistence.room.Room; **import** android.support.v7.app.AppCompatActivity; **import** android.os.Bundle; **import** android.view.View; **import** android.widget.EditText; **import** android.widget.Toast;  **import** com.mobsandgeeks.saripaar.ValidationError; **import** com.mobsandgeeks.saripaar.Validator; **import** com.mobsandgeeks.saripaar.annotation.Email; **import** com.mobsandgeeks.saripaar.annotation.NotEmpty; **import** com.mobsandgeeks.saripaar.annotation.Url;  **import** java.util.List;  **import** sv.edu.udb.reciclaje.data.AppDatabase; **import** sv.edu.udb.reciclaje.data.RecyclingCenter; **import** sv.edu.udb.reciclaje.data.RecyclingCenterDAO;  **public class** AddRecyclingCenterActivity **extends** AppCompatActivity **implements** Validator.ValidationListener {   @NotEmpty  EditText **name**;  @Url  EditText **imageSiteURL**;  EditText **address**;  @Email  EditText **email**;  @NotEmpty  EditText **latitude**;  @NotEmpty  EditText **longitude**;   Validator **validator**;   @Override  **protected void** onCreate(Bundle savedInstanceState) {    **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_add\_recycling\_center***);   **validator** = **new** Validator(**this**);  **validator**.setValidationListener(**this**);    **name** = findViewById(R.id.***name***);  **imageSiteURL** = findViewById(R.id.***imagesiteurl***);  **address** = findViewById(R.id.***address***);  **email** = findViewById(R.id.***email***);  **latitude** = findViewById(R.id.***latitude***);  **longitude** = findViewById(R.id.***longitude***);  }   **public void** add(View view){   **validator**.validate();   }   **private void** insert(){  AppDatabase database = Room.*databaseBuilder*(**this**, AppDatabase.**class**, **"mydb"**)  .allowMainThreadQueries()  .build();   RecyclingCenterDAO recyclingCenterDAO = database.getRecyclingCenterDAO();   RecyclingCenter recyclingCenter = **new** RecyclingCenter();  recyclingCenter.setName(**name**.getText().toString());  recyclingCenter.setEmail(**email**.getText().toString());  recyclingCenter.setAddress(**address**.getText().toString());  recyclingCenter.setLatitude(Double.*parseDouble*(**latitude**.getText().toString()));  recyclingCenter.setLongitude(Double.*parseDouble*(**longitude**.getText().toString()));  recyclingCenter.setImageSiteUrl(**imageSiteURL**.getText().toString());   recyclingCenterDAO.insert(recyclingCenter);   finish();  }   @Override  **public void** onValidationSucceeded() {  insert();  }   @Override  **public void** onValidationFailed(List<ValidationError> errors) {  **for** (ValidationError error : errors) {  View view = error.getView();  String message = error.getCollatedErrorMessage(**this**);   *// Display error messages ;)* **if** (view **instanceof** EditText) {  ((EditText) view).setError(message);  } **else** {  Toast.*makeText*(**this**, message, Toast.***LENGTH\_LONG***).show();  }  }  }   } |

**IV Ejercicio complementario**

|  |
| --- |
| Proyecto: **Sistema de cotizaciones móvil.** |
| Localmente vía **~~GreenDAO~~ Room**, almacenará su base de datos de clientes. Debe guardar nombre, dirección, teléfono, email.  Listará categorías de productos, al seleccionarlo desplegará los productos con sus precios. Esta información será accedida vía webservice.  El usuario vía GreenDAO podrá guardar la información de la cotización. Deberá guardar el nombre de usuario de aplicación, el correo(destino), los IDs de los item en la cotización individualmente, cantidad, precio sugerido (webservice) precio ofertado (manual) y el total.  Finalmente, la información será enviada en formato json vía petición POST hacia el webservice, el cual enviará un correo detallado hacia el cliente.  La aplicación pedirá usuario y password (Esto estará habilitado al final del taller). |

Almacenar la lista de clientes utilizando Room . Debe guardar nombre, dirección, teléfono, email. Utilice el formulario para agregar usuarios, además, debe crear un RecyclerView para mostrar el listado de los clientes agregados. Al seleccionar un cliente, debe permitirle eliminarlo/modificarlo si se considera necesario.