Reporte de Primera Evaluación Práctica

Tracking de productos congelados - Pingüi

Paralelo: 1 Grupo: 3

Materia: Aplicaciones Móviles y Servicios telemáticos

Profesora: Msig. Adriana Collaguazo Jaramillo

Fecha de inicio: 28/6/2019

Fecha de finalización: 30/6/2019

* Viviana Mero
* Allan Orellana

# Resultados de las actividades

Para el desarrollo de la activada se

## Clase Main Activity

La clase Main Activity será la primera que visualizará el usuario cuando inicie la aplicación móvil. Esta actividad será la encargada de realizar un inicio de sesión, para lo cual pregunta por credenciales creadas en la base de datos externa en herokuapp. Si las credenciles coinciden, el API externo cederá un token de acceso y la app avanzará a la siguiente activadad: Menu. Caso contario se mostrará un mensaje en forma de Toast indicando el mensaje de error: “Credenciales Incorrectas”.

Se muestra a continuación las vistas:

### Vista clase Main

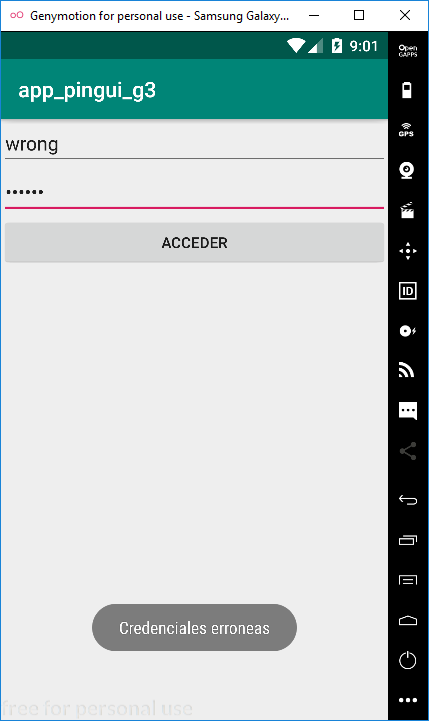
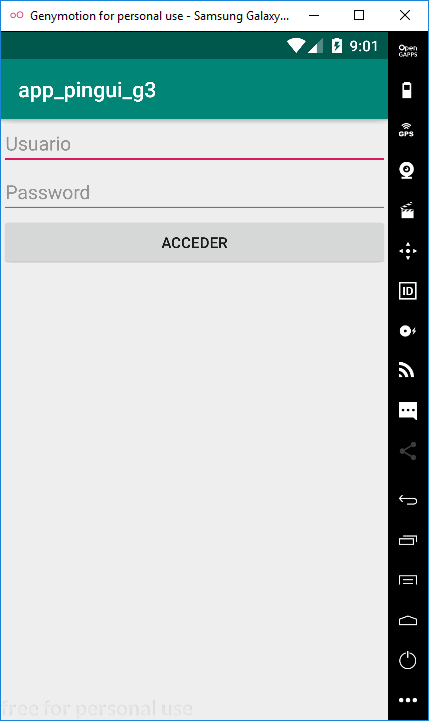
La vista consta con dos EditText que servirán para ingresar las credenciales de usuario y contraseña, además de que cuentan con sus respectivas “hints”. También se añade un botón con la palabra “Acceder” el mismo que servirá para ejecutar la acción de verificar las credenciales en el API. En el caso de que el par de credenciales no se encuentre en la base de datos de herokuapp se mostrará un Toast con el mensaje “Credenciales Incorrectas” caso contrario la aplicación avanzará a la activad Menú

Ilustración 1 En la imagen de la izquierda se observa

XML Actividad Main

Se presenta el código XML de la activad Main con sus respectivas explicaciones:

**Se inicia el código XML con los parámetros por defecto que crea Android Stududio. Estos se los dejará igual**

<?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=".MainActivity">

**Se inicia un contenedor que mantendrá un orden relativo en la visualización de los demás widgets. En este caso, se optó por usar un LinealLayot vertical que se ajuste al espacio de la pantalla.**  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
**Se añade el primer EditText, este servira para que se ingrese la credencial de “usuario”. Se coloca como id = “txtUser” y se coloca como hint = “Usuario”. Las demas configuraciones las dejamos por defecto.**  
 <EditText  
 android:id="@+id/txtUser"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="Usuario"  
 android:inputType="textPersonName"  
 android:password="false" />  
**Se añade el segundo EditText, este servira para que se ingrese la credencial de “password”. Se coloca como id = “txtPas”, se coloca como hint = “Password” y se coloca el parametr de password = true. Las demás configuraciones las dejamos por defecto.**  
 <EditText  
 android:id="@+id/txtPass"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="Password"  
 android:inputType="textPersonName"  
 android:password="true" />  
**Finalmente se añade el boton que realizará la accion una vez que el usuario haya ingresado las credeciales en los dos EditText superiors. Se coloca como id = button, text = Acceder y onClick =login donde “login” es el nombre de la function a la que hara refencia una vez precionado por el usuario.**  
 <Button  
 android:id="@+id/button"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:onClick="login"  
 android:text="Acceder" />  
 </LinearLayout>  
</androidx.constraintlayout.widget.ConstraintLayout>

### **Código activadad Main**

**Declaraciones del paquete de donde pertenece la clase, importaciones de las diferentes clases y funciones a usar en esta clase Main**

package com.grupo2.app\_pingui\_g3;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import com.android.volley.Request;  
import com.android.volley.RequestQueue;  
import com.android.volley.Response;  
import com.android.volley.VolleyError;  
import com.android.volley.toolbox.JsonObjectRequest;  
import com.android.volley.toolbox.Volley;  
  
import org.json.JSONException;  
import org.json.JSONObject;  
  
import java.util.HashMap;  
import java.util.Map;  
  
public class MainActivity extends AppCompatActivity {

**Se inicia declarando los widgets que se utilizaran del XML**  
 TextView usuario;  
 TextView pass;  
 RequestQueue queue;  
 Button start;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
**Se instancia una clase de Request Queue, esta misma sera la encargada de enviar las peticiones http y también de recibirlas**  
 queue = Volley.*newRequestQueue*(this);

**Se asigna el valor de los widgets**  
 usuario = (TextView) findViewById(R.id.*txtUser*);  
 pass = (TextView) findViewById(R.id.*txtUser*);  
 start = (Button)findViewById(R.id.*button*);  
  
 }  
**Se crea la function login, misma que esta asociada a las accion del boton “Acceder” de la via Main**  
 public void login(View view){

**Se declara la direccion url a donde mandar la peticion post**  
 String url= "https://amstdb.herokuapp.com/db/nuevo-jwt";

**Se crea un HashMap con los parametros enviados en la peticion POST. Estos se intancia despues eun JsonObject**  
 Map<String, String> params = new HashMap<>();  
 params.put("username",usuario.getText().toString());  
 params.put("password",pass.getText().toString());  
 JSONObject a = new JSONObject(params);

**Se instance un JsonObjectRequest que contiene la logica de la petcion, es decir: Se trata de una peticion POST con los parametros creados en el blcque de codigo anterior y se lo realiza a la direccion web declarada en url.**   
 JsonObjectRequest request = new JsonObjectRequest(Request.Method.*POST*, url, a, new Response.Listener<JSONObject>() {

**Si la peticion es favorable se ejecuta la function onResponse**  
 @Override  
 public void onResponse(JSONObject response) {  
 try {

**La respuesta del API de herokuapp es un JSON y se guarada en response. Debido a que es un json obtenemos el valor asocaido a la clave “token” y se lo almacena en la variable token.**  
 String token = response.getString("token");

**Se intancia una nueva activad, en este caso a ser favorable se incia la acitivdad Menu y se pasa como Extra la variable token, con el fin de que sea utilizada por esta nueva activadad**  
 Intent intent = new Intent(getApplicationContext(), Menu.class);  
 intent.putExtra("token", token);  
 startActivity(intent);

**En caso de existir alguna falla en este proceso pero si existio una respuesta favorable por el API, caera en este catch y mostrará en un Toast el mensaje = “Mal Parseado”**  
 } catch (JSONException e) {  
 Toast a = Toast.*makeText*(getApplicationContext(), "Mal parseado", Toast.*LENGTH\_LONG*);  
 a.show();  
 //e.printStackTrace();  
 }  
 }

**En caso de no haber respuesta favorable por el API se ejecutara la funcioin onErrorResponse**  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {

**Se muestra en un toast el mensaje =” Credenciales erróneas”**  
 System.*out*.println("nada");  
 Toast a = Toast.*makeText*(getApplicationContext(), "Credenciales erroneas", Toast.*LENGTH\_LONG*);  
 a.show();  
 }  
 });

**Finalmente la lógica colocada en la variable request es añadida al objeto RequestQueue en queue y se envía la pericion http.**  
 queue.add(request);  
  
 }  
}

## Actividad Menu

## Vista Menu

La vista de la actividad Menu esta conformada únicamente por tres botones que servirán para mostrar las temperaturas en la base de datos de herokuapp, mostrar los recorridos de la base de datos de herokuapp y un botón que servirá para salir de la aplicación móvil. Estos están predispuestos verticalmente y con los textos: Temperatura, Recorrirdo y Salir respectivamente.

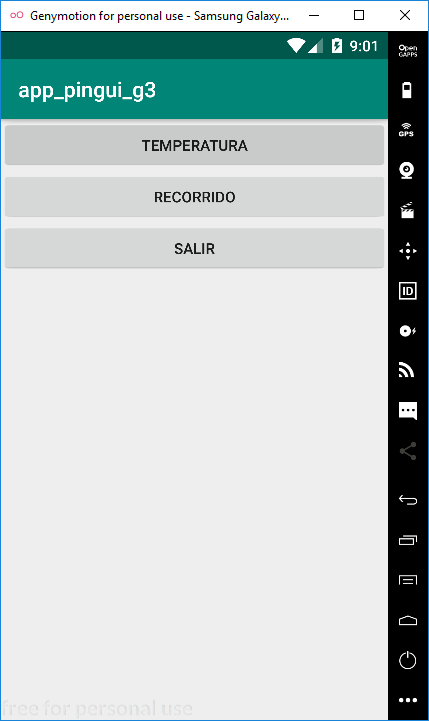


Ilustración 2 Vista de la activad Menú que consiste en tres botones: Temperatura, Recorrido y Salir

### XML de la activad Menú

**Se inicia el código XML con los parámetros por defecto que crea Android Stududio. Estos se los dejará igual**

<?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=".Menu">  
**Se inicia un contenedor que mantendrá un orden relativo en la visualización de los demás widgets. En este caso, se optó por usar un LinealLayot vertical que se ajuste al espacio de la pantalla.**  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">

**Se añade el primer botón que servirá para iniciar la activad Temperatura. Se coloca como id = buttonTemp, text= Temperatura y el evento onClick se asocia a la function = irTemp**  
  
 <Button  
 android:id="@+id/buttonTemp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:onClick="irTemp"  
 android:text="Temperatura" />

**Se añade el segundo botón que servirá para iniciar la activad Recorrido. Se coloca como id = buttonRec, text= Recorrido y el evento onClick se asocia a la función= irRec**  
  
 <Button  
 android:id="@+id/buttonRec"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:onClick="irRec"  
 android:text="Recorrido" />  
**Se añade el segundo botón que servirá para cerrar la aplicación móvil. Se coloca como id = buttoSalir, text= Salir y el evento onClick se asocia a la function: salir**  
 <Button  
 android:id="@+id/buttonSalir"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:onClick="salir"  
 android:text="Salir" />  
 </LinearLayout>  
</androidx.constraintlayout.widget.ConstraintLayout>

### Código actividad Menu

**Declaraciones del paquete de donde pertenece la clase, importaciones de las diferentes clases y funciones a usar en esta clase Menu**

package com.grupo2.app\_pingui\_g3;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.Toast;  
  
public class Menu extends AppCompatActivity {

**Se declara la variable token para poder ser utilziada en diferentes funciones**  
 String token;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_manu*);

**Se a signa el valor de la variable token como el valor obtendio de los extras, el mismo que fue pasado por la clase Main**   
 token = getIntent().getExtras().getString("token");  
 }

**Se crea la funcion irTemp que esta asociada al botón Temperatura. En esta función se inicializa la actividad Temperatura pasandole como extra la variable token. De manera analoga con la función irRec que está asociada al botón Recorrido e inicializa la actividad Recorrido.**  
 public void irTemp(View v){  
 Intent i = new Intent(getApplicationContext(), Temperatura.class);  
 i.putExtra("token", token);  
 startActivity(i);  
 }  
  
 public void irRec(View v){  
 Intent i = new Intent(getApplicationContext(), Rec.class);  
 i.putExtra("token", token);  
 startActivity(i);  
 }

**Se crea la función salir que está asociada al botón Salir. En esta función se finaliza la aplicación móvil y además muestra el mensaje = “App finalizada” en un Toast.**  
 public void salir(View v){  
 this.finishAffinity();  
 Toast toast = Toast.*makeText*(getApplicationContext(), "APP finalizada", Toast.*LENGTH\_LONG*);  
 toast.show();  
 }  
  
}

## Actividad Temperartura

### Vista Temperatura

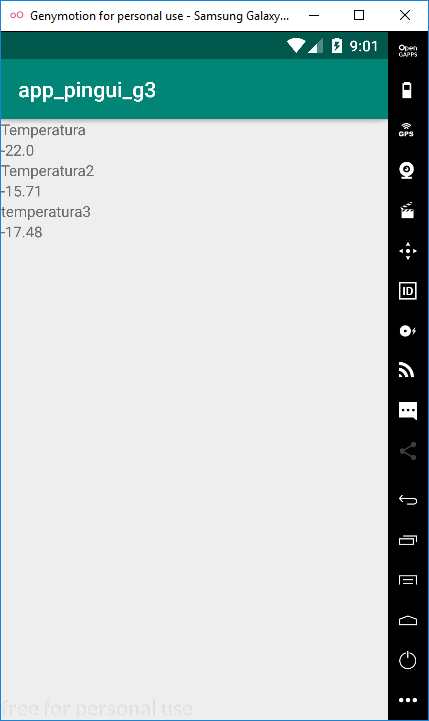
La vista Temperatura cuenta únicamente con TextViews que servirán para mostrar los valores de temperatura obtenidos mediante el API de herokuapp. Se presentan de manera vertical.

Ilustración 3 Vista de actividad Temperatura. Se muestra tres valores de temperatura almacenados en la base de datos externa de herokuapp

### XML actividad Temperatura

**Se inicia el código XML con los parámetros por defecto que crea Android Stududio. Estos se los dejará igual**

<?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=".Temperatura">  
**Se inicia un contenedor que mantendrá un orden relativo en la visualización de los demás widgets. En este caso, se optó por usar un LinealLayot vertical que se ajuste al espacio de la pantalla.**  
 <LinearLayout  
 android:id="@+id/lineal"  
  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">

**Se crea un conjunto de pares de TextViews que servirá para mostrar en pantalla la etiqueta de la temperatura y su valor respectivamente. En este caso existe un total de tres pares, cada uno de los TextViews que presentaran el valor de temperatura tiene un id accendente.**  
  
 <TextView  
 android:id="@+id/textView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Temperatura" />  
  
 <TextView  
 android:id="@+id/textTemp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content" />  
  
 <TextView  
 android:id="@+id/textView5"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Temperatura2" />  
  
 <TextView  
 android:id="@+id/texttemp1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textView7"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="temperatura3" />  
  
 <TextView  
 android:id="@+id/texttemp3"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 </LinearLayout>  
</androidx.constraintlayout.widget.ConstraintLayout>

### Codigo actividad Temperatura

**Declaraciones del paquete de donde pertenece la clase, importaciones de las diferentes clases y funciones a usar en esta clase Main**

package com.grupo2.app\_pingui\_g3;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.LayoutInflater;  
import android.widget.LinearLayout;  
import android.widget.TextView;  
  
import com.android.volley.AuthFailureError;  
import com.android.volley.Request;  
import com.android.volley.RequestQueue;  
import com.android.volley.Response;  
import com.android.volley.VolleyError;  
import com.android.volley.toolbox.JsonObjectRequest;  
import com.android.volley.toolbox.Volley;  
  
import org.json.JSONArray;  
import org.json.JSONException;  
import org.json.JSONObject;  
  
import java.lang.reflect.Array;  
import java.util.ArrayList;  
import java.util.HashMap;  
import java.util.Map;  
  
public class Temperatura extends AppCompatActivity {

**Se inicia declarando los widgets que se utilizaran del XML**  
 String token;  
 RequestQueue queue;  
 TextView temp;  
 TextView temp2;  
 TextView temp3;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_temp*);

**Se obtiene el valor de la raible token de los extras.**  
 token = getIntent().getExtras().getString("token");

**Se instancia una clase de Request Queue, esta misma sera la encargada de enviar las peticiones http y también de recibirlas**  
 queue = Volley.*newRequestQueue*(this);

**Se asigna valor a los widgets declarados**  
 temp = findViewById(R.id.*textTemp*);  
 temp2 = findViewById(R.id.*texttemp1*);  
 temp3 = findViewById(R.id.*texttemp3*);

**Appenas se cree esta actividad se ejecuran las siguintestes funciones. Las misma que se encargarán de ejecutar la peticion http**  
 getTemps1();  
 getTemps2();  
 getTemps3();  
 }

public void getTemps1(){

**Se asigna la direccio url**  
 String url= "https://amstdb.herokuapp.com/db/registroDeFrios/1";  
**Se instancia la peticion con el metedo Get, y sin parametros**  
 JsonObjectRequest request = new JsonObjectRequest(Request.Method.GET, url, null, new Response.Listener<JSONObject>() {  
 @Override  
 public void onResponse(JSONObject response) {  
 try {

**Cuendo la respuesta es exitosa se obtiene un JSON y se extrae el valor de la clave “temperature” y se lo asigna al correspondiente textView**  
 System.out.println(response);  
 String value = response.getString("temperatura");  
   
 System.out.println(value);  
 temp.setText(value);  
  
 } catch (JSONException e) {  
 System.out.println("Error parseo");  
 e.printStackTrace();  
 }  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {

**Si la respuesta no es favorable se crea un log en la consola**  
 System.out.println("nada");  
 System.out.println(error);  
 }  
 }){

**Debido a que el API utiliza el token de acceso, este es colocado como un parametron en la cabecera de la peticion GET**  
 @Override  
 public Map<String,String> getHeaders() throws AuthFailureError {  
 Map<String, String> params = new HashMap<String,String>();  
 params.put("Authorization", "JWT " + token);  
 System.out.println(token);  
 return params;  
 }  
 };

**Finalmente se envia la peticion a la cola para que esta sea gestionada y despachada**  
 queue.add(request);  
 }

**Las siguientes dos funciones hacen lo mismo pero con otro valor de temperature y es asignado a otro textView**  
 public void getTemps2(){  
 String url= "https://amstdb.herokuapp.com/db/registroDeFrios/2";  
  
 JsonObjectRequest request = new JsonObjectRequest(Request.Method.GET, url, null, new Response.Listener<JSONObject>() {  
 @Override  
 public void onResponse(JSONObject response) {  
 try {  
 System.out.println(response);  
 String value = response.getString("temperatura");  
 //String a[] = value.split(",");  
 System.out.println(value);  
 temp2.setText(value);  
  
 } catch (JSONException e) {  
 System.out.println("Error parseo");  
 e.printStackTrace();  
 }  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {  
 System.out.println("nada");  
 System.out.println(error);  
 }  
 }){  
 @Override  
 public Map<String,String> getHeaders() throws AuthFailureError {  
 Map<String, String> params = new HashMap<String,String>();  
 params.put("Authorization", "JWT " + token);  
 System.out.println(token);  
 return params;  
 }  
 };  
 queue.add(request);  
 }  
  
 public void getTemps3(){  
 String url= "https://amstdb.herokuapp.com/db/registroDeFrios/3";  
  
 JsonObjectRequest request = new JsonObjectRequest(Request.Method.GET, url, null, new Response.Listener<JSONObject>() {  
 @Override  
 public void onResponse(JSONObject response) {  
 try {  
 System.out.println(response);  
 String value = response.getString("temperatura");  
 //String a[] = value.split(",");  
 System.out.println(value);  
 temp3.setText(value);  
  
 } catch (JSONException e) {  
 System.out.println("Error parseo");  
 e.printStackTrace();  
 }  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {  
 System.out.println("nada");  
 System.out.println(error);  
 }  
 }){  
 @Override  
 public Map<String,String> getHeaders() throws AuthFailureError {  
 Map<String, String> params = new HashMap<String,String>();  
 params.put("Authorization", "JWT " + token);  
 System.out.println(token);  
 return params;  
 }  
 };  
 queue.add(request);  
 }  
   
}

## Actividad Recorrido

### Vista

Solo muestra los valores almacenados en la base de datos externa

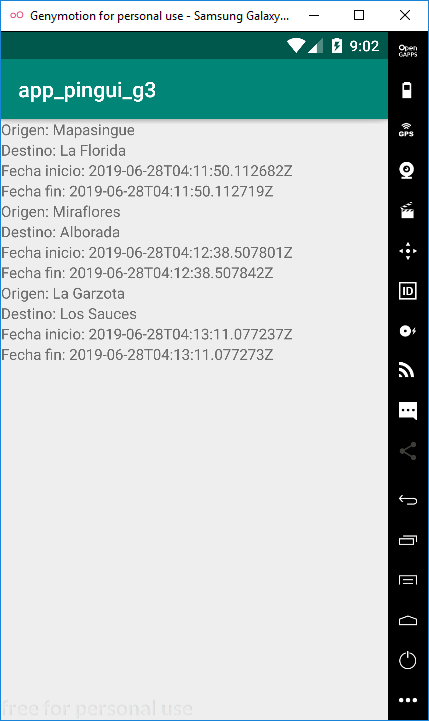


Ilustración 4Vista Recorrido, solo muestra los datos de la base de datos externa

**NOTA: El siguiente código es análogo al código de la clase Temperatura, la misma explicación sirve para ambos casos.**

## XML Recorrido

<?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=".Recorrido">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
  
 <TextView  
 android:id="@+id/textRec"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textdest"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textf1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textf2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textRec2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textDest2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textf12"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textf22"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textRec3"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textdest3"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textf13"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 <TextView  
 android:id="@+id/textf23"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="TextView" />  
  
 </LinearLayout>  
  
</androidx.constraintlayout.widget.ConstraintLayout>

### Código Recorrido

package com.grupo2.app\_pingui\_g3;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
import android.widget.TextView;  
  
import com.android.volley.AuthFailureError;  
import com.android.volley.Request;  
import com.android.volley.RequestQueue;  
import com.android.volley.Response;  
import com.android.volley.VolleyError;  
import com.android.volley.toolbox.JsonObjectRequest;  
import com.android.volley.toolbox.Volley;  
  
import org.json.JSONException;  
import org.json.JSONObject;  
  
import java.util.HashMap;  
import java.util.Map;  
  
public class Recorrido extends AppCompatActivity {  
 String token;  
 RequestQueue queue;  
 TextView rec;  
 TextView dest;  
 TextView f1;  
 TextView f2;  
  
 TextView rec2;  
 TextView dest2;  
 TextView f12;  
 TextView f22;  
  
 TextView rec3;  
 TextView dest3;  
 TextView f13;  
 TextView f23;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_rec);  
 token = getIntent().getExtras().getString("token");  
 queue = Volley.newRequestQueue(this);  
 rec = findViewById(R.id.textRec);  
 dest = findViewById(R.id.textdest);  
 f1 = findViewById(R.id.textf1);  
 f2 = findViewById(R.id.textf2);  
  
 rec2 = findViewById(R.id.textRec2);  
 dest2 = findViewById(R.id.textDest2);  
 f12 = findViewById(R.id.textf12);  
 f22 = findViewById(R.id.textf22);  
  
 rec3 = findViewById(R.id.textRec3);  
 dest3 = findViewById(R.id.textdest3);  
 f13 = findViewById(R.id.textf13);  
 f23 = findViewById(R.id.textf23);  
  
 getTemps1();  
 getTemps2();  
 getTemps3();  
 }  
  
 public void getTemps1(){  
 String url= " https://amstdb.herokuapp.com/db/recorrido/1";  
  
 JsonObjectRequest request = new JsonObjectRequest(Request.Method.GET, url, null, new Response.Listener<JSONObject>() {  
 @Override  
 public void onResponse(JSONObject response) {  
 try {  
 System.out.println(response);  
 String v1 = response.getString("origen");  
 String v2 = response.getString("destino");  
 String v3 = response.getString("fecha\_inicio");  
 String v4 = response.getString("fecha\_fin");  
  
  
 //String a[] = value.split(",");  
  
 rec.setText("Origen: " + v1);  
 dest.setText("Destino: " + v2);  
 f1.setText("Fecha inicio: " + v3);  
 f2.setText("Fecha fin: " + v4);  
 } catch (JSONException e) {  
 System.out.println("Error parseo");  
 e.printStackTrace();  
 }  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {  
 System.out.println("nada");  
 System.out.println(error);  
 }  
 }){  
 @Override  
 public Map<String,String> getHeaders() throws AuthFailureError {  
 Map<String, String> params = new HashMap<String,String>();  
 params.put("Authorization", "JWT " + token);  
 System.out.println(token);  
 return params;  
 }  
 };  
  
 queue.add(request);  
 }  
  
  
  
 public void getTemps2() {  
 String url = " https://amstdb.herokuapp.com/db/recorrido/2";  
  
 JsonObjectRequest request = new JsonObjectRequest(Request.Method.GET, url, null, new Response.Listener<JSONObject>() {  
 @Override  
 public void onResponse(JSONObject response) {  
 try {  
 System.out.println(response);  
 String v1 = response.getString("origen");  
 String v2 = response.getString("destino");  
 String v3 = response.getString("fecha\_inicio");  
 String v4 = response.getString("fecha\_fin");  
  
  
 rec2.setText("Origen: " + v1);  
 dest2.setText("Destino: " + v2);  
 f12.setText("Fecha inicio: " + v3);  
 f22.setText("Fecha fin: " + v4);  
 } catch (JSONException e) {  
 System.out.println("Error parseo");  
 e.printStackTrace();  
 }  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {  
 System.out.println("nada");  
 System.out.println(error);  
 }  
 }) {  
 @Override  
 public Map<String, String> getHeaders() throws AuthFailureError {  
 Map<String, String> params = new HashMap<String, String>();  
 params.put("Authorization", "JWT " + token);  
 System.out.println(token);  
 return params;  
 }  
 };  
  
 queue.add(request);  
 }  
 public void getTemps3() {  
 String url = " https://amstdb.herokuapp.com/db/recorrido/3";  
  
 JsonObjectRequest request = new JsonObjectRequest(Request.Method.GET, url, null, new Response.Listener<JSONObject>() {  
 @Override  
 public void onResponse(JSONObject response) {  
 try {  
 System.*out*.println(response);  
 String v1 = response.getString("origen");  
 String v2 = response.getString("destino");  
 String v3 = response.getString("fecha\_inicio");  
 String v4 = response.getString("fecha\_fin");  
  
  
 rec3.setText("Origen: " + v1);  
 dest3.setText("Destino: " + v2);  
 f13.setText("Fecha inicio: " + v3);  
 f23.setText("Fecha fin: " + v4);  
 } catch (JSONException e) {  
 System.out.println("Error parseo");  
 e.printStackTrace();  
 }  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {  
 System.out.println("nada");  
 System.out.println(error);  
 }  
 }) {  
 @Override  
 public Map<String, String> getHeaders() throws AuthFailureError {  
 Map<String, String> params = new HashMap<String, String>();  
 params.put("Authorization", "JWT " + token);  
 System.out.println(token);  
 return params;  
 }  
 };  
  
 queue.add(request);  
 }  
}

## Permisos en Manifest

**Para poder hacer peticiones http externas es necesario añadir el permiso a internet.**

<uses-permission  
 android:name="android.permission.INTERNET"></uses-permission>

**Para poder cerrar la aplicación es necesario añadir el siguiente permiso**

<uses-permission  
 android:name="android.permission.KILL\_BACKGROUND\_PROCESSES"></uses-permission>

# Link Github

https://github.com/allantaws/AMST\_1E\_G3.git