

**Escuela Superior Politécnica del Litoral**

**Aplicaciones Móviles y Servicios Telemáticos**

# Informe de la Segunda Evaluación Práctica

**Msig. Adriana Collaguazo**

**Grupo 5**

**Abel Silva**

**Cesar Navas**

**Bolívar Núñez**

**17/01/2020**

**Descripción del Desarrollo de la segunda evaluación práctica**

1. **Pantalla principal**

**Còdigo JAVA**

En el MainActivity se realizo la pantalla principal que contiene los datos del grupo, titulo. Tambien presenta un EditText que permite al usuario ingresar una palabra para realizar una búsqueda de ese super héroe en la base de datos al pulsar el Button BuscarHeroes, el cual implementa la acción de enviar la palabra ingresada a la activity RegistroHero e iniciarla.

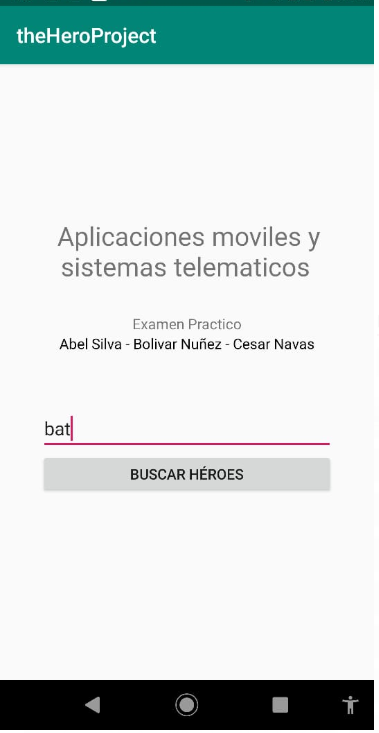
public class MainActivity extends AppCompatActivity {  
  
 public Button btn\_Buscar\_Heroes ;  
 public EditText edtNombre;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 btn\_Buscar\_Heroes = findViewById(R.id.*btn\_busqueda*);  
 edtNombre = findViewById(R.id.*edtNombre*);  
 }  
  
 public void Resultados(View v){  
 Intent intent = new Intent(this , RegistroHero.class);  
 intent.putExtra("NOMBRE",edtNombre.getText().toString());  
 startActivity(intent);  
 }  
}

**Còdigo XML**

Para la pantalla de presentación incial se utilizo TextView para presentar información como el título, los nombres de los integrantes, un EditText para que el usuario ingrese aquella palabra semejante al nombre de héroe que desee buscar y un botón que permite implementar la acción con el método Resultados.

<?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">  
  
 <TextView  
 android:id="@+id/textView2"  
 android:layout\_width="292dp"  
 android:layout\_height="92dp"  
 android:text="Aplicaciones moviles y sistemas telematicos "  
 android:textSize="25dp"  
 android:textAlignment="center"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.526"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintRight\_toRightOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.296" />  
  
 <TextView  
 android:id="@+id/textView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Examen Practico"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView2" />  
  
 <TextView  
 android:id="@+id/textView3"  
 android:layout\_width="280dp"  
 android:layout\_height="33dp"  
 android:text="Abel Silva - Bolivar Nuñez - Cesar Navas"  
 android:textAlignment="center"  
 android:textColor="@android:color/black"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView" />  
  
 <EditText  
 android:id="@+id/textView4"  
 android:layout\_width="283dp"  
 android:layout\_height="45dp"  
 android:layout\_marginTop="36dp"  
 android:hint="Busqueda por nombre"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView3" />  
  
 <Button  
 android:id="@+id/btn\_busqueda"  
 android:layout\_width="283dp"  
 android:layout\_height="42dp"  
 android:text="Buscar Héroes"  
 android:onClick="Resultados"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView4" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>

**Ejecución**



1. **Ventana de resultados**

La activity RegistroHero permite presentar en forma de una lista de botones los nombres de todos los héroes que contengan la palabra ingresada como referencia. El usuario puede elegir el héroe al cual se quiere referir y es enviado a la siguiente ventana y activity de perfilHero.

**Còdigo JAVA**

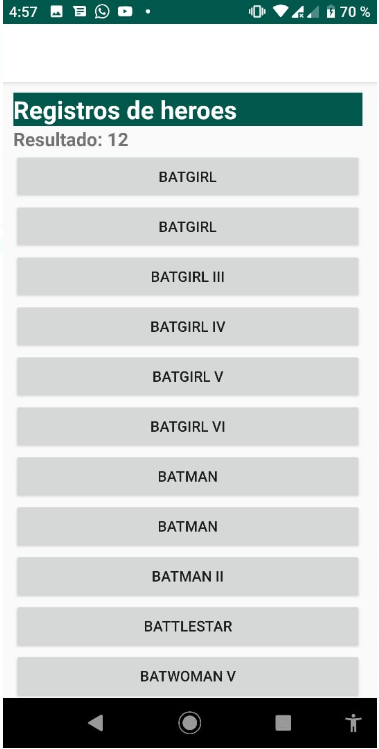
public class RegistroHero extends AppCompatActivity {  
  
 public BarChart graficoBarras;  
 private RequestQueue ListaRequest = null;  
 private String token = "3429546483754576";  
  
 //Generar un token propio  
 private String nombre;  
 private LinearLayout contHeroe;  
 private RegistroHero contexto;  
 private TextView txtResultado;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_registro\_hero*);  
  
 ListaRequest = Volley.*newRequestQueue*(this);  
 contexto = this;  
 nombre = getIntent().getStringExtra("NOMBRE");  
  
 solicitarDatos(nombre);  
 }  
  
 */\*\*  
 \* Función solicitarDatos();  
 \* Esta función se encarga de llamar al RESTAPI obtener la información solicitada, que en este caso es a la informacion  
 \* de todos los heroes que contengan la palabra de la varibale name dentro en sus nombres. Esta funcion  
 \* llama otra uncion: mostrarHeroes().  
 \*/* public void solicitarDatos(String name){  
 String url\_registros = "https://superheroapi.com/api/"+token+"/search/"+name;  
 JsonObjectRequest requestRegistros =  
 new JsonObjectRequest( Request.Method.*GET*, url\_registros, null, new Response.Listener<JSONObject>() {  
 @Override  
 public void onResponse(JSONObject response) {  
 mostrarHeroes(response);  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {  
 System.*out*.println(error.getMessage());  
 }  
 }  
 ){ @Override  
 public Map<String, String> getHeaders() {  
 Map<String, String> params = new HashMap<>();  
 params.put("Authorization", "JWT " + token);  
 return params;  
 }  
 };  
 ListaRequest.add(requestRegistros);  
 }  
  
 */\*\*  
 \* Función mostrarHeroes();  
 \* Esta función obtiene la información de la base de datos del API y se de filtrar los datos de  
 \* los nombres de los heroes para mostrarlos en una lista al usuario, en forma de botones para que  
 \* el usuario elija a cual se está refiriendo. Cada boton implementa la accion de redirigir al usuario a  
 \* la activity perfilHero y envia datos como nombre y id del heroe.  
 \*/* private void mostrarHeroes(JSONObject hero){  
  
 JSONArray resultados;  
 JSONObject listHero;  
 txtResultado = findViewById(R.id.*txtResultado*);  
  
 contHeroe = findViewById(R.id.*Heroe*);  
 try  
 {  
 resultados = (JSONArray) hero.get("results");  
 String cantHeroes="Resultado: "+String.*valueOf*(resultados.length());  
 txtResultado.setText(cantHeroes);  
  
 for (int i = 0; i < resultados.length(); i++) {  
 listHero = (JSONObject) resultados.get(i);  
 TextView btn = new Button(RegistroHero.this);  
 final String nombre;  
 final String ID;  
 nombre = listHero.getString("name");  
 ID = listHero.getString("id");  
  
 btn.setText(nombre);  
 System.*out*.println(resultados);  
 btn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Toast.*makeText*(getApplicationContext(), "Heroe confirmado", Toast.*LENGTH\_SHORT*).show();  
 Intent intent = new Intent(getApplicationContext(), perfilHero.class);  
 intent.putExtra("name", nombre);  
 intent.putExtra("id", ID);  
 startActivity(intent);  
 }  
 });  
  
 contHeroe.addView(btn);  
 }  
 } catch (JSONException e) {  
 e.printStackTrace();  
 System.*out*.println("error");  
 }  
 }  
  
  
}

**Código XML**

En el codigo xml se utilizo un scrollView para que no haya perdida de datos y que el usuario pueda desplazarse por toda la información que se presente en la pantalla. Todos los botones que se vayan creando con los resultados de la búsqueda son agregados al LinearLayout Heroe.

<?xml version="1.0" encoding="utf-8"?>  
<ScrollView  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent" android:fillViewport="true"  
 android:scrollbars = "vertical" android:scrollbarStyle="insideInset"  
 >  
 <LinearLayout  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:orientation="vertical"  
 android:padding="10dp"  
 android:id="@+id/contenedor">  
  
 <TextView  
 android:id="@+id/textView2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Registros de heroes"  
 android:background="@color/colorPrimaryDark"  
 android:textColor="@color/colorPrimary"  
 android:textSize="24sp"  
 android:textStyle="bold" />  
 <TextView  
 android:id="@+id/txtResultado"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Registros de heroes"  
 android:textSize="18sp"  
 android:textStyle="bold" />  
  
 <LinearLayout  
 android:id="@+id/Heroe"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical" >  
  
 </LinearLayout>  
 </LinearLayout>  
</ScrollView>

**Ejecución**



1. **Grafica de estadísticas del heroe**

La activity perfilHero permite presentar un gráfico de barras con toda la información de las estadísticas del héroe, de igual forma permite presentar el nombre de héroe y el nombre completo de la persona.

**Código JAVA**

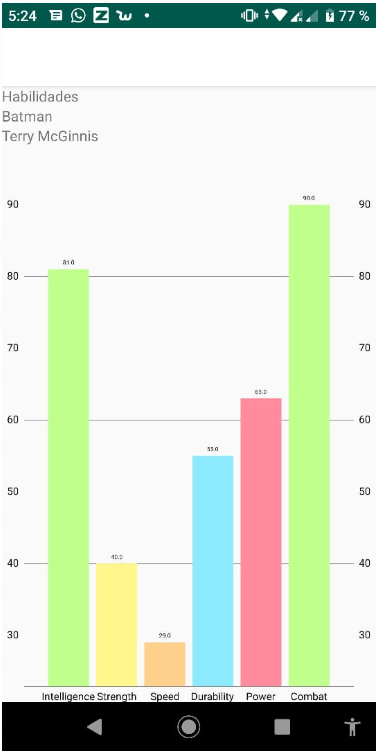
public class perfilHero extends AppCompatActivity {  
  
 public BarChart graficoBarras;  
 private RequestQueue ListaRequest = null;  
 private String token = "3429546483754576";  
 private perfilHero contexto;  
 private String name;  
 private String id;  
 private TextView txtNombreCompleto, txtNombreHero;  
 private String[] valuesX= new String[]{"Intelligence","Strength","Speed","Durability", "Power", "Combat"};  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_perfil\_hero*);  
  
 //setTitle("Grafico de barras");  
 ListaRequest = Volley.*newRequestQueue*(this);  
 contexto = this;  
  
 name = getIntent().getStringExtra("name");  
 id = getIntent().getStringExtra("id");  
  
  
  
 /\* GRAFICO \*/  
 this.iniciarGrafico();  
 this.solicitarDatos(name, id);  
 }  
  
 */\*\*  
 \* Función IniciarGrafico();  
 \* Obtiene el grafico desde la vista (layout) y coloca las propiedades de inicio. Propiedades como: tamaño,  
 \* descripción, animaciones, títulos colocadas en esta función. (Estas propiedades son finales y no  
 \* cambian por ninguna condición.)  
 \*/* public void iniciarGrafico() {  
 graficoBarras = findViewById(R.id.*barChart*);  
 graficoBarras.getDescription().setEnabled(false);  
 graficoBarras.setMaxVisibleValueCount(60);  
 graficoBarras.setPinchZoom(false);  
 graficoBarras.setDrawBarShadow(false);  
 graficoBarras.setDrawGridBackground(false);  
 XAxis xAxis = graficoBarras.getXAxis();  
 xAxis.setPosition(XAxis.XAxisPosition.*BOTTOM*);  
 xAxis.setDrawGridLines(false);  
 graficoBarras.getAxisLeft().setDrawGridLines(false);  
 graficoBarras.animateY(1500);  
 graficoBarras.getLegend().setEnabled(false);  
 }  
  
 */\*\*  
 \* Función solicitarDatos();  
 \* Esta función se encarga de llamar al RESTAPI obtener la información solicitada del heroe basados en el id,  
 \* que es enviada a la funcion de mostrarResultados() Y actualizarGrafico(). Tambien presenta el nombre del heroe  
 \* en el TextView txtNombreHero.  
 \*/* public void solicitarDatos(String nombre, String Id){  
 txtNombreHero = findViewById(R.id.*txtNombreHero*);  
  
 txtNombreHero.setText(nombre);  
 System.*out*.println(Id);  
  
 String url\_registros = "https://superheroapi.com/api/"+token+"/"+Id;  
 JsonObjectRequest requestRegistros =  
 new JsonObjectRequest( Request.Method.*GET*, url\_registros, null, new Response.Listener<JSONObject>() {  
 @Override  
 public void onResponse(JSONObject response) {  
 mostrarResultados(response);  
 actualizarGrafico(response);  
 }  
 }, new Response.ErrorListener() {  
 @Override  
 public void onErrorResponse(VolleyError error) {  
 System.*out*.println(error.getMessage());  
 }  
 }  
 ){ @Override  
 public Map<String, String> getHeaders() {  
 Map<String, String> params = new HashMap<>();  
 params.put("Authorization", "JWT " + token);  
 return params;  
 }  
 };  
 ListaRequest.add(requestRegistros);  
 }  
  
 */\*\*  
 \* Función mostrarResultados();  
 \* Esta función obtiene la información de la bibliografia del heroe. Se encarga de filtrar los datos de  
 \* de nombre completo del heroe para mostrarlos en el TextView txtNombreCompleto.  
 \*/* private void mostrarResultados(JSONObject hero){  
 JSONObject biografia;  
 String fullname;  
 txtNombreCompleto = findViewById(R.id.*txtNombreCompleto*);  
 try{  
 biografia = (JSONObject) hero.get("biography");  
 fullname = biografia.getString("full-name");  
 txtNombreCompleto.setText(fullname);  
  
 } catch (JSONException e) {  
 e.printStackTrace();  
 System.*out*.println("error");  
 }  
 }  
  
 */\*\*  
 \* Función actualizarGrafico();  
 \* Esta función actualiza y obtiene los datos de estadisticas del heroe para actualizar el grafico en el app.  
 \* Llamamos a la función llenarGrafico() ingresando el nuevo dato obtenido.  
 \*/* private void actualizarGrafico(JSONObject hero){  
  
 JSONObject powerStats;  
  
 float intelligence;  
 float Strength;  
 float Speed;  
 float Durability;  
 float Power;  
 float Combat;  
  
 ArrayList<BarEntry> dato\_hero = new ArrayList<>();  
  
 try{  
  
 powerStats = (JSONObject) hero.get("powerstats");  
  
 intelligence = powerStats.getInt("intelligence");  
 dato\_hero.add(new BarEntry(0, intelligence));  
  
 Strength = powerStats.getInt("strength");  
 dato\_hero.add(new BarEntry(1, Strength));  
  
 Speed = powerStats.getInt("speed");  
 dato\_hero.add(new BarEntry(2, Speed));  
  
 Durability = powerStats.getInt("durability");  
 dato\_hero.add(new BarEntry(3, Durability));  
  
 Power = powerStats.getInt("power");  
 dato\_hero.add(new BarEntry(4, Power));  
  
 Combat = powerStats.getInt("combat");  
 dato\_hero.add(new BarEntry(5, Combat));  
  
 } catch (JSONException e) {  
 e.printStackTrace();  
 System.*out*.println("error");  
 }  
 System.*out*.println(dato\_hero);  
 llenarGrafico(dato\_hero);  
 }  
  
 */\*\*  
 \* Función llenarGrafico();  
 \* Esta función ocurrirá el Live Reload durante llenarGrafico() actualizaremos el grafico actual con el  
 \* nuevo dato encontrado. Pero también ejecutaremos un Hilo (como Runnable) para ejecutar una acción  
 \* de fondo, donde se volverá a buscar un nuevo dato (si es que este existe) al llamar nuevamente a la  
 \* función solicitarDatos() iniciando el ciclo..  
 \*/* private void llenarGrafico(ArrayList<BarEntry> dato\_hero){  
 BarDataSet heroDataSet;  
  
 if ( graficoBarras.getData() != null && graficoBarras.getData().getDataSetCount() > 0) {  
 heroDataSet = (BarDataSet) graficoBarras.getData().getDataSetByIndex(0);  
 heroDataSet.setValues(dato\_hero);  
 graficoBarras.getData().notifyDataChanged();  
 graficoBarras.notifyDataSetChanged();  
  
 } else {  
 heroDataSet = new BarDataSet(dato\_hero, "Data Set");  
 heroDataSet.setColors(ColorTemplate.*VORDIPLOM\_COLORS*);  
 heroDataSet.setDrawValues(true);  
 ArrayList<IBarDataSet> dataSets = new ArrayList<>();  
  
 graficoBarras.getXAxis().setValueFormatter(new IndexAxisValueFormatter(valuesX));  
 dataSets.add(heroDataSet);  
 BarData data = new BarData(dataSets);  
 graficoBarras.setData(data); graficoBarras.setFitBars(true);  
 }  
 graficoBarras.invalidate();  
 final Handler handler = new Handler();  
 final Runnable runnable = new Runnable() {  
 @Override  
 public void run() {  
 solicitarDatos(name, id);  
 }  
 };  
 handler.postDelayed(runnable, 3000);  
 }  
  
}

**Código XML**

El archivo xml de perfilHero contiene TextView para presentar el nombre del héroe y su nombre completo. También contiene un BarChart utilizando la librería de MPAndroid que contendrá la grafica de barra con los datos que se ingresen en ella.

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout 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:orientation="vertical"  
 tools:context=".perfilHero">  
  
 <TextView  
 android:id="@+id/txtHabilidades"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Habilidades" />  
 <TextView  
 android:id="@+id/txtNombreHero"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="nombre" />  
 <TextView  
 android:id="@+id/txtNombreCompleto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="nombre completo" />  
  
 <com.github.mikephil.charting.charts.BarChart  
 android:id="@+id/barChart"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
 </com.github.mikephil.charting.charts.BarChart>  
</LinearLayout>

**Ejecución**



* **Link de repositorio**

https://github.com/bolisteward/AMST\_2EP\_Grupo5\_Nunez\_Silva\_Navas.git