Desarrollo de aplicaciones avanzadas con Android

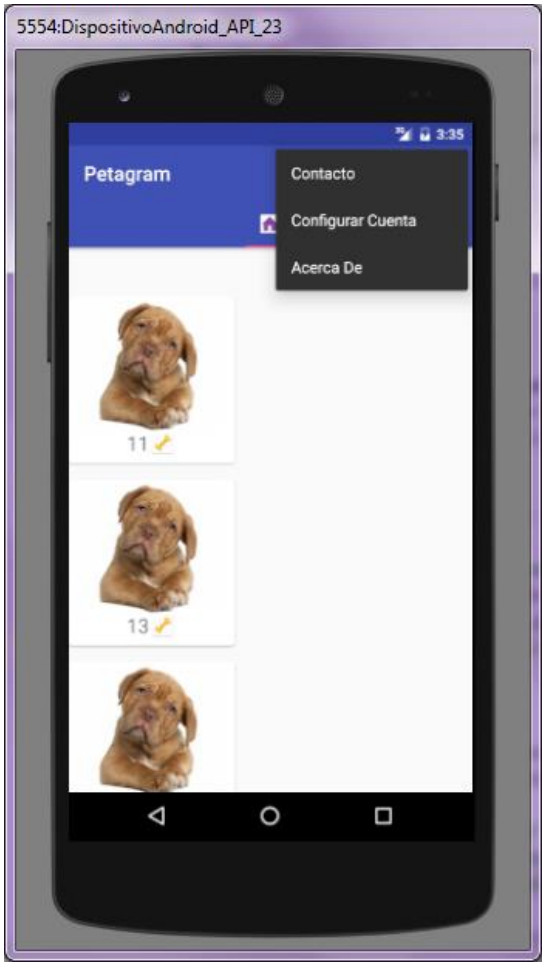
Integrando Web Services

Por: Nestor Estrada Olivares

Uso de Endpoints de Instagram

En la pantalla principal se muestra la nueva opción del menú “Configurar Cuenta”, la cual nos mostrará la pantalla donde se introducirá el nombre de la cuenta compartida por sandbox que podrá ser mostrada en la aplicación.

Cabe mencionar que las imágenes mostradas en la primer pantalla son las que se generaron en el ejercicio de Base de Datos, debido al cambio del POJO , la imagen no cambia y queda asignada una por default. Además solo se muestra un fragment, el fragment faltante se genera cuando se presionar el botón de guardar cuenta, siempre y cuando se obtenga un resultado satisfactorio.



Para este ejercicio ocupe 2 Endpoints, los cuales describo a continuación:

GET /users/search

https://api.instagram.com/v1/users/search?q=jack&access_token=ACCESS-TOKEN

Este Endpoint se utilizó para poder obtener el Id de la cuenta que se introduce en la pantalla de configurar cuenta:

En ConstantesRestApi, se declararon las constantes para formar el URL:

```
ndPointsApi.java × | ConstantesRestApi.java × | MainActivity.java × | MascotasFragment.java × | MascotaAdaptador.java ×

package mx.alnegasoft.mascotamd.restApi;

import ...

/**
 * Created by NEO on 11/08/2016.
 */
public class ConstantesRestApi {

    public static final String VERSION                    = "/v1/";
    public static final String ROOT_URL                   = "https://api.instagram.com" + VERSION;
    public static final String ACCESS_TOKEN               = "3648605337.bef215d.5a54ae5f797a4ec2b2b2fef2df1f5832";
    public static final String KEY_ACCESS_TOKEN           = "&access_token=";
    public static final String KEY_USERS_SEARCH           = "users/search?";


    public static final String KEY_ACCESS_TOKEN_MEDIA_USER  = "access_token=";
    public static final String KEY_GET_RECENT_MEDIA_USER_ID = "users/{userid}/media/recent/?";
    //public static final String KEY_GET_RECENT_MEDIA_USER_ID = "users/3648605337/media/recent/?";
    public static final String URL_GET_RECENT_MEDIA_USER_ID = KEY_GET_RECENT_MEDIA_USER_ID + KEY_ACCESS_TOKEN_MEDIA_USER + ACCE
```

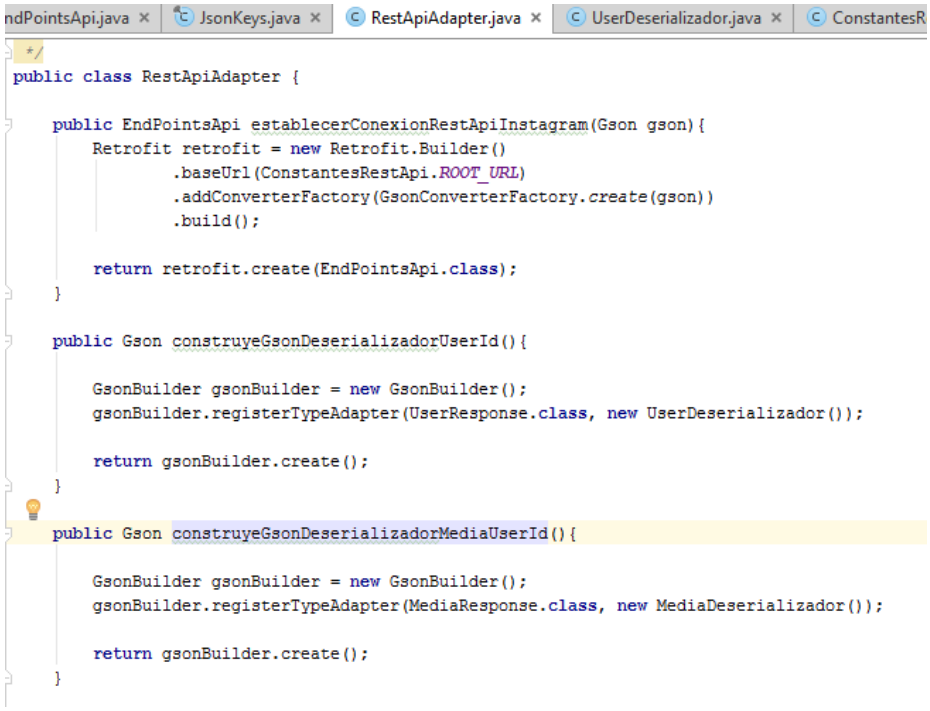
Debido a que el nombre de la cuenta es un texto variable, no se podía completar la URL con solo constantes, por lo que se utilizó el @Query para anexar los datos faltantes:

```
EndPointsApi.java × | ConstantesRestApi.java × | MediaResponse.java × | MainActivity.java × | Mascot

package mx.alnegasoft.mascotamd.restApi;

import ...

/**
 * Created by ABITA on 12/08/2016.
 */
public interface EndPointsApi {


    @GET(ConstantesRestApi.KEY_USERS_SEARCH)
    Call<UserResponse> getUserId(@Query("q") String q ,@Query("access_token") String at);
```

En UserDeserializador, se obtiene el Id y su imagen de perfil:

```
ndPointsApi.java × | JsonKeys.java × | RestApiAdapter.java × | UserDeserializador.java × | ConstantesRestApi.java ×

    }

    private ArrayList<Mascota> deserializarUsuarioDeJson(JsonArray userResponseData){
        ArrayList<Mascota> usuarios = new ArrayList<>();

        for (int i = 0; i <userResponseData.size() ; i++) {
            JsonObject userResponseDataObject       = userResponseData.get(i).getAsJsonObject();
            //JsonObject userJson                         = contactoResponseDataObject.getAsJsonObject(JsonKeys.USER);
            String id                                 = userResponseDataObject.get(JsonKeys.USER_RESPONSE_ID).getAsString();
            String profilePicture                     = userResponseDataObject.get(JsonKeys.USER_ID_PROFILE_PICTURE).getAsStr

            //usuarios.add(id);
            Mascota usuarioActual = new Mascota();

            usuarioActual.setId(id);
            usuarioActual.setUrlFotoPerfil(profilePicture);

            usuarios.add(usuarioActual);

        }
        return usuarios;
    }
```

En RestApiAdapter, se ve la aplicación de Retrofit:



```
ndPointsApi.java × | JsonKeys.java × | RestApiAdapter.java × | UserDeserializador.java × | ConstantesR
 */
public class RestApiAdapter {

    public EndPointsApi establecerConexionRestApiInstagram(Gson gson){
        Retrofit retrofit = new Retrofit.Builder()
                .baseUrl(ConstantesRestApi.ROOT_URL)
                .addConverterFactory(GsonConverterFactory.create(gson))
                .build();

        return retrofit.create(EndPointsApi.class);
    }

    public Gson construyeGsonDeserializadorUserId(){

        GsonBuilder gsonBuilder = new GsonBuilder();
        gsonBuilder.registerTypeAdapter(UserResponse.class, new UserDeserializador());

        return gsonBuilder.create();
    }

    public Gson construyeGsonDeserializadorMediaUserId(){

        GsonBuilder gsonBuilder = new GsonBuilder();
        gsonBuilder.registerTypeAdapter(MediaResponse.class, new MediaDeserializador());

        return gsonBuilder.create();
    }
```

En ConfigurarCuenta, se ejecuta la función obtenerUserId() al presionar el botón Guardar Cuenta, la función obtiene los datos para enviarlos al MainActivity donde se asignaran al fragment correspondiente:

```
ava × | RestApiAdapter.java × | UserDeserializador.java × | ConstantesRestApi.java × | MediaResponse.java × | ConfigurarCuenta.java ×
    public void obtenerUserId() {
        RestApiAdapter restApiAdapter = new RestApiAdapter();
        Gson gsonUserId = restApiAdapter.construyeGsonDeserializadorUserId();
        EndPointsApi endPointsApi = restApiAdapter.establecerConexionRestApiInstagram(gsonUserId);

        android.support.design.widget.TextInputLayout tilUsuario;
        tilUsuario = (android.support.design.widget.TextInputLayout) findViewById(R.id.tilUsuario);
        final String nombreUsuario = tilUsuario.getEditText().getText().toString();
        Toast.makeText(ConfigurarCuenta.this, nombreUsuario, Toast.LENGTH_SHORT).show();

        Call<UserResponse> userResponseCall = endPointsApi.getUserId(nombreUsuario ,ConstantesRestApi.ACCESS_TOKEN );

        userResponseCall.enqueue(new Callback<UserResponse>() {
            @Override
            public void onResponse(Call<UserResponse> call, Response<UserResponse> response) {
                UserResponse  userResponse = response.body();
                usuarios =  userResponse.getUsuarios();


                Toast.makeText(getBaseContext(), "Configurar Cuenta Usuario: " + usuarios.get(0).getId().toString(), Toast.LENG
                Toast.makeText(getBaseContext(), "Configurar Cuenta Foto Perfil: " + usuarios.get(0).getUrlFotoPerfil().toStrir
                Toast.makeText(getBaseContext(), "Nombre Cuenta: " + nombreUsuario, Toast.LENGTH_SHORT).show();
                //mostrarContactosRV();

                Intent intent = new Intent(getBaseContext() ,MainActivity.class);
                intent.putExtra("userId",usuarios.get(0).getId().toString());
                intent.putExtra("fotoPerfil",usuarios.get(0).getUrlFotoPerfil().toString());
                intent.putExtra("nombreCuenta",nombreUsuario);
                startActivity(intent);
```

GET /users/user-id/media/recent

https://api.instagram.com/v1/users/{user-id}/media/recent/?access_token=ACCESS-TOKEN

Este Endpoint se utilizó para poder obtener el último contenido subido por la cuenta que fue seleccionada en opción configurar cuenta.

La secuencia es la misma que en el Endpoint anterior, el principal cambio es que el parámetro se pasa utilizando @Path en lugar de @Query, la diferencia es que @Path se ocupa cuando el valor requerido esta solicitado por un "=" y el @Query cuando esta dentro del URL como una variable dentro de {variable}.

ndPointsApi.java × | JsonKeys.java × | RestApiAdapter.java × | UserDeserializador.java × | ConstantesR

```
public interface EndPointsApi {


    @GET(ConstantesRestApi.KEY_USERS_SEARCH)
    Call<UserResponse> getUserId(@Query("q") String q ,@Query("access_token") String at);

    @GET(ConstantesRestApi.URL_GET_RECENT_MEDIA_USER_ID)
    Call<MediaResponse> getRecentMediaUser(@Path("userid") String userid);
```

ConstantesRestApi.java × | MediaResponse.java × | ConfigurarCuenta.java × | MascotaAdaptador.java × | MediaDeserializador.java ×

```
private ArrayList<Mascota> deserializarMediaDeJson(JsonArray mediaResponseData){
    ArrayList<Mascota> mascotas = new ArrayList<>();

    for (int i = 0; i <mediaResponseData.size() ; i++) {
        JsonObject mediaResponseDataObject      = mediaResponseData.get(i).getAsJsonObject();
        JsonObject mediaJson                    = mediaResponseDataObject.getAsJsonObject(JsonKeys.USER);
        String id                               = mediaJson.get(JsonKeys.USER_ID).getAsString();
        String profilePicture                   = mediaJson.get(JsonKeys.USER_ID_PROFILE_PICTURE).getAsString();
        String nombre                           = mediaJson.get(JsonKeys.USER_FULLNAME).getAsString();

        JsonObject imageJson                    = mediaResponseDataObject.getAsJsonObject(JsonKeys.MEDIA_IMAGES);
        JsonObject stdResolutionJson            = imageJson.getAsJsonObject(JsonKeys.MEDIA_STANDARD_RESOLUTION);
        String urlFoto                          = stdResolutionJson.get(JsonKeys.MEDIA_URL).getAsString();

        JsonObject likesJson                    = mediaResponseDataObject.getAsJsonObject(JsonKeys.MEDIA_LIKES);
        int likes                               = likesJson.get(JsonKeys.MEDIA_LIKES_COUNT).getAsInt();

        Mascota mascotaActual = new Mascota();

        mascotaActual.setId(id);
        mascotaActual.setNombre(nombre);
        mascotaActual.setUrlFoto(urlFoto);
        mascotaActual.setLikes(likes);
        mascotaActual.setUrlFotoPerfil(profilePicture);

        mascotas.add(mascotaActual);

    }
    return mascotas;
```

Debido a que estamos utilizando los fragments, era necesario pasar los datos como argumentos, por lo que se tuvo que hacer la implementación antes de asignar el fragment al array de fragments que se ocupa en el ViewPager:

```
private ArrayList<Fragment> agregarFragments(){
    ArrayList<Fragment> fragments = new ArrayList<>();
    fragments.add(new MascotasFragment());

    Intent intent = getIntent();
    Bundle extras = intent.getExtras();
    if (extras != null) {
        userId = (String) extras.get("userId");
        fotoPerfil = (String) extras.get("fotoPerfil");
        nombreCuenta = (String) extras.get("nombreCuenta");
        Toast.makeText(MainActivity.this, "Main Activity UserId: "+ userId, Toast.LENGTH_SHORT).show();

        if (!userId.equals("0")) {

            PerfilFragment perfilFragment =  new PerfilFragment();
            Bundle arguments = new Bundle(3);

            arguments.putString("userId", userId);
            arguments.putString("fotoPerfil", fotoPerfil);
            arguments.putString("nombreCuenta", nombreCuenta);

            perfilFragment.setArguments(arguments);

            fragments.add(perfilFragment);
        }

    }
```

En el fragment se hace el resto de la implementación, donde se ocupan los valores enviados desde la pantalla de configurar cuenta, para colocar el nombre de la cuenta y su foto de perfil, junto con su contenido reciente.

```
lainActivity.java × | PerfilFragment.java × | UserDeserializador.java × | ConstantesRestApi.java × | MediaResponse.java ×

public void obtenerMediaUserId() {
    RestApiAdapter restApiAdapter = new RestApiAdapter();
    Gson gsonMediaUserId = restApiAdapter.construyeGsonDeserializadorMediaUserId();
    EndPointsApi endPointsApi = restApiAdapter.establecerConexionRestApiInstagram(gsonMediaUserId);

    String userId="0";
    if (getArguments() != null) {
        userId = getArguments().get("userId").toString();
        fotoPerfil = getArguments().get("fotoPerfil").toString();
        nombreCuenta = getArguments().get("nombreCuenta").toString();

        circularImageView = (com.mikhaellopez.circularimageview.CircularImageView) v.findViewById(R.id.circularImageView);
        Picasso.with(getContext())
                .load(fotoPerfil)
                .placeholder(R.drawable.perro1)
                .into(circularImageView);

        tvNombreCuenta = (TextView) v.findViewById(R.id.tvNombreCuenta);
        tvNombreCuenta.setText(nombreCuenta);
        Toast.makeText(getContext(), "Main Activity UserId: "+ userId, Toast.LENGTH_SHORT).show();
        Call<MediaResponse> mediaResponseCall = endPointsApi.getRecentMediaUser(userId);
        //Call<MediaResponse> mediaResponseCall = endPointsApi.getRecentMediaUser();

        mediaResponseCall.enqueue(new Callback<MediaResponse>() {
            @Override
            public void onResponse(Call<MediaResponse> call, Response<MediaResponse> response) {
                MediaResponse mediaResponse = response.body();
                mascotas = mediaResponse.getMascotas();
                inicializarAdaptador();
```

Como resultado regresamos a nuestra pantalla principal, donde se generó el fragment donde tenemos la información solicitada:

Video de la Aplicación:

https://youtu.be/bj_IbeikpdM