TUTORIAL 3 CONEXIÓN E INTERPRETACION DE DATOS CON JSON

Pedro Góngora Soria

Descripción

- Hay muchas plataformas web que ofrecen un servicio de consultas de datos, pero la manera o la base de datos que usan pueden ser un problema a la hora de captar esos datos con tu web o app.
- Entonces para esto se usa Json que es un formato de texto ligero para intercambio de datos.
- En mi problema en concreto necesitaba pasar un array de muchísima información a mi app y no sabia como obtenerla , y para ello use el siguiente código y la interpretación de json.

Descripción

- En este caso, voy a usar el ejemplo de mi propia aplicación, de como necesitaba obtener los datos de una persona, aparte de los personales, necesitaba obtener el arrayList de toda su lista de comidas que había agregado a la aplicación. La cual se obtiene por un campo que indica que esa comida pertenece a este usuario.
- Se supone que como es la pantalla después del login ya hemos obtenido su identificador como he enseñado en el tutorial anterior, realizando la consulta y devolviendo los datos necesarios.

Código

He añadido en azul la parte importante del código, en mi caso en la parte que se crea la activity para obtener todo ahí.

Como se puede ver, llamo a JsonTask, que es una consulta Json, que se ejecuta en la siguiente URL.

Dicha url tiene que ofrecer un servicio de intercambio de datos Json, en este caso lo había implementado yo así que no hay problema.

De ahí captara los datos que interpretaremos en las siguientes transparencias.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_usuario);
    objGlobal = (globalClass)getApplicationContext();

    usuario = (TextView)findViewById(R.id.TextV_usuario);
    kcal = (TextView)findViewById(R.id.TextV_Kcal);
    kcalDiarias = (TextView)findViewById(R.id.TextV_KcalDiarias);

    prot = (TextView)findViewById(R.id.TextV_protMax);
    protDiarias = (TextView)findViewById(R.id.TextV_protDiarias);

    hidra = (TextView)findViewById(R.id.TextV_hidraMax);
    hidraDiarias = (TextView)findViewById(R.id.TextV_hidraDiarios);

    grasas = (TextView)findViewById(R.id.TextV_grasasMax);
    grasasDiarias = (TextView)findViewById(R.id.TextV_grasasDiarios);

    txtJson = (TextView)  findViewById(R.id.tv_pruebas);



    //clase global
    //referencia al objeto global de mi aplicacion
    //objGlobal.set_KcalDiarias(400);

    ///////////

    Intent intent = getIntent();
    String name = objGlobal.nombreUsuarioT;
     user = objGlobal.UserName;
     password = objGlobal.pass;

     IdUser = objGlobal.idUser;
    int  calorias = objGlobal.KcalT;
    int  protRecibidas = objGlobal.proteinasT;
    int  hidraRecibidas = objGlobal.HidratosT;
    int grasasRecibidas = objGlobal.grasasT;
    objGlobal.comidas.clear();
    new Usuario.JsonTask().execute("http://www.miweb.com/obtenerComidas.php?id="+IdUser);
    kcalDiarias.setText(objGlobal.getKcalDiarias()+"");
    protDiarias.setText(objGlobal.getProtDiarias()+"");
    hidraDiarias.setText(objGlobal.getHidraDiarias()+"");
    grasasDiarias.setText(objGlobal.getGrasasDiarias()+"");

    objGlobal.idUser =IdUser;
```

Código

Ahora habría que crear la función de interpretación de dichos Datos.

Tendriamos que declarar una variable global llamada Data del tipo String para ir almacenando los datos y ahora pasarlos a tipo Json:

```
String data;
String dataParsed = "";
```

Y después de obtener el mensaje JSON, si el formato es correcto podríamos obtener los datos como se indica, usando una vez mas la palabra clave para obtenerlos.

```
private void InterpretarDatos() throws JSONException {

    String nombre;
    String kcal;
    String proteinas;
    String hidratos;
    String grasas;
    String gramos;
    objGlobal = (globalClass)getApplicationContext();
    JSONArray JA = new JSONArray(data);
    for (int i=0; i<JA.length(); i++) {
        JSONObject JO = (JSONObject) JA.get(i);
        nombre=(String)JO.get("nombre");
        kcal=(String)JO.get("kcal");
        proteinas=(String)JO.get("proteinas");
        hidratos=(String)JO.get("hidratos");
        grasas=(String)JO.get("grasas");
        gramos=(String)JO.get("gramos");

        objGlobal.añadirComida(nombre,Integer.parseInt(kcal),Integer.parseInt(proteinas),Integer.parseInt(hidratos),Integer.parseInt(grasas),Integer.parseInt(gramos));

    }


}
```

Código

Que devolvería nuestro Script o Servicio web ? Pues algo de este estilo :

[{"id":"1","pertenece":"2","nombre":"macarrones","kcal":"300","proteinas":"15","hidratos":"80","grasas":"10","gramos":"100"},{"id":"2","pertenece":"2","nombre":"yougurt dieta","kcal":"100","proteinas":"20","hidratos":"10","grasas":"0","gramos":"100"},{"id":"3","pertenece":"2","nombre":"pizza","kcal":"900","proteinas":"50","hidratos":"120","grasas":"40","gramos":"100"},{"id":"4","pertenece":"2","nombre":"vaso de leche","kcal":"30","proteinas":"5","hidratos":"20","grasas":"0","gramos":"100"},{"id":"5","pertenece":"2","nombre":"zumo","kcal":"150","proteinas":"2","hidratos":"30","grasas":"2","gramos":"100"},{"id":"7","pertenece":"2","nombre":"tazon cereales","kcal":"100","proteinas":"100","hidratos":"11","grasas":"200","gramos":"100"},{"id":"8","pertenece":"2","nombre":"pepinillos","kcal":"200","proteinas":"10","hidratos":"20","grasas":"40","gramos":"100"},{"id":"23","pertenece":"2","nombre":"macarrones","kcal":"400","proteinas":"20","hidratos":"45","grasas":"15","gramos":"100"}]

Como se puede ver es un Array que contiene todas las comidas que tenia dicho usuario registradas.

Y se puede acceder a sus campos usando las palabras clave grasas, gramos, pertenece, nombre ….

Todos los servicios de este tipo , tienen que tener este formato ya que es la manera de interpretarlo JSON y si no daría error.

Código

Por ultimo añado el código para poder usar Json que debe de ir en toda Actividad que se cree o bien crear Una clase que lo implemente :

```
protected String doInBackground(String... params) {


    HttpURLConnection connection = null;
    BufferedReader reader = null;

    try {
        URL url = new URL(params[0]);
        connection = (HttpURLConnection) url.openConnection();
        connection.connect();


        InputStream stream = connection.getInputStream();

        reader = new BufferedReader(new InputStreamReader(stream));

        StringBuffer buffer = new StringBuffer();
        String line = "";

        while ((line = reader.readLine()) != null) {
            buffer.append(line+"\n");
            Log.d( tag: "Response: ",  msg: "> " + line);   //here u ll get whole response..... :-)

        }

        return buffer.toString();


    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
```

```
            e.printStackTrace();
        } finally {
            if (connection != null) {
                connection.disconnect();
            }
            try {
                if (reader != null) {
                    reader.close();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        return null;
    }

    @Override
    protected void onPostExecute(String result) {
        super.onPostExecute(result);
        if (pd.isShowing()){
            pd.dismiss();
        }
        data = result;
        try {
            InterpretarDatos();
        } catch (JSONException e) {
            e.printStackTrace();
        }
        txtJson.setText(dataParsed);

    }
}
```

Código

Como se puede ver le llega como parámetro una lista de String, por lo que se puede usar varias URL para obtener los datos.
En este caso solo me interesaba una así que por eso hago uso de params[0].

La siguiente parte es crear un objeto de la clase StringBuffer, que como su nombre indica actúa de Buffer para escribir y otro de lo que vaya a leer (BufferedReader), y un aplicar Buffer.append .

Este método actualiza el valor del objeto que invocó el método. El método toma boolean, char, int, long, Strings, etc.

Lo que vaya leyendo hay que darle formato y estará en el Reader, y pasaría al StringBuffer.
Por ultimo lo pasamos a String y lo devolvemos.

Código

La ultima parte del código es la que se encargaría de una vez obtenido el contenido, usarlo por ejemplo asignándolo a TextView, para ver si recibimos el contenido bien o mal y hacer la comprobación.

Y si no como siempre, en caso de haber
Una excepción, avisar de ella.

```
    @Override
    protected void onPostExecute(String result) {
        super.onPostExecute(result);
        if (pd.isShowing()){
            pd.dismiss();
        }
        data = result;
        try {
            InterpretarDatos();
        } catch (JSONException e) {
            e.printStackTrace();
        }
        txtJson.setText(dataParsed);

    }
}
```