Parsing JSON avec Jackson sous Android

Par Nazim Benbourahla - nbenbourahla

Date de publication : 5 décembre 2011

Dans ce tutoriel, nous allons aborder le parsing de fichiers **JSON** (JavaScript Object Notation) à l'aide de la bibliothèque **Jackson**.

1 - Qu'est-ce que JSON ?

JSON est un format léger d'échange de données. Il est facile à écrire et à lire, facilement analysable. C'est un format totalement indépendant de tout langage, ce qui en fait un langage d'échange de données idéal.

Il se base sur deux structures :

- une collection de couple “Nom / Valeur” ;
- une liste de valeurs ordonnées.

2 - Pourquoi Jackson ?

Pour cet article, nous allons utiliser la bibliothèque **Jackson**. Vous allez sûrement vous demander “**Pourquoi Jackson ?**”. Tout simplement parce qu'à mon avis, c'est la meilleure bibliothèque de parsing **JSON** en général et aussi sous Android. Ce **benchmark** vous aidera à en être convaincu.
Jackson est une bibliothèque :

- de Streaming (Lecture / Ecriture) ;
- rapide ;
- puissante ;
- possédant aucune dépendance ;
- open source ;
- configurable à souhait.

Vous pouvez obtenir plus d'informations sur le site officiel de **Jackson**.

3 - Mise en place du projet

Pour notre projet, nous allons avoir besoin d'un exemple de fichier **JSON**. Voici notre petit exemple qui sera disponible sur : **users.json**.

```
{
    "Users": [
        {
            "firstname": "Nazim",
            "lastname": "Benbourahla",
            "login": "n_benbourahla",
            "twitter": "@n_benbourahla",
            "web": ""
        },
        {
            "firstname": "Tutos",
            "lastname": "android",
            "login": "Tutos-android",
            "twitter": "",
            "web": "www.tutos-android.com"
        }
    ]
}
```

Nous allons créer un projet Android avec les caractéristiques suivantes :

- nom du projet : TutosAndroidJsonParser ;
- version du SDK : 2.2 ;
- nom de l'application : JSON Parser ;
- package : com.tutos.android.json ;
- Activité : JsonParserMainActivity.

Nous allons créer un dossier **lib** à la racine de notre projet dans lequel nous allons mettre les *jars* de **Jackson** qui sont disponible **ici**.
Sans oublier de rajouter les jars au Build Path du projet.

Pour commencer nous allons créer notre vue, qui sera composée d'un bouton et d'un texte pour afficher le **JSON**.

```
<?xmlversion="1.0"encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    >

<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Parser le fichier JSON"
    android:id="@+id/startParsing"
/>

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/jsonDisplay"
    />
</LinearLayout>
```

Puis nous allons créer nos modèles, correspondant à notre objet User passé dans le JSON.
Voici notre classe User.

```
package com.tutos.android.json.model;

public class User {
    private String firstname;
    private String lastname;
    private String login;
    private String twitter;
    private String web;

    public User() {
    super();
    this.firstname = "";
    this.lastname = "";
    this.login = "";
    this.twitter = "";
    this.web = "";
    }

    public User(String firstName, String lastName, String login,
                String twitter, String web) {
    super();
    this.firstname = firstName;
    this.lastname = lastName;
    this.login = login;
    this.twitter = twitter;
    this.web = web;
    }

    public String getFirstname() {
        returnfirstname;
    }

    public void setFirstname(String firstName) {
        this.firstname = firstName;
    }

    public String getLastname() {
```

```
        return lastname;
    }

    public void setLastname(String lastName) {
        this.lastname = lastName;
    }

    public String getLogin() {
        return login;
    }

    public void setLogin(String login) {
        this.login = login;
    }

    public String getTwitter() {
        return twitter;
    }

    public void setTwitter(String twitter) {
        this.twitter = twitter;
    }

    public String getWeb() {
        return web;
    }

    public void setWeb(String web) {
        this.web = web;
    }

}
```

Puis la liste des utilisateurs, qui correspond tout simplement à une HashMap de correspondance.

```
package com.tutos.android.json.model;

import java.util.ArrayList;
import java.util.HashMap;

public class Users extendsHashMap<String, ArrayList<User>> {

    private static final longserialVersionUID = 1L;

}
```

Une fois notre modèle prêt, nous allons créer une classe qu'on nommera **UserController**. Cette classe s'occupera de télécharger et parser notre JSON.

```
package com.tutos.android.json.model;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.ArrayList;

import org.codehaus.jackson.JsonFactory;
import org.codehaus.jackson.JsonParseException;
import org.codehaus.jackson.JsonParser;
import org.codehaus.jackson.map.ObjectMapper;

import android.os.Environment;

public class UsersController {
```

```
private static final String DL_URL = "";http://www.tutos-android.com/JSON/users.json

private ObjectMapper objectMapper = null;
private JsonFactory jsonFactory = null;
private JsonParser jp = null;
private ArrayList<User> userList = null;
private Users users = null;
private File jsonOutputFile;
private File jsonFile;

public UsersController() {
objectMapper = newObjectMapper();
jsonFactory = newJsonFactory();
}

public void init() {
downloadJsonFile();
try{
    jp = jsonFactory.createJsonParser(jsonFile);
    users = objectMapper.readValue(jp, Users.class);
    userList = users.get("Users");
} catch(JsonParseException e) {
    e.printStackTrace();
} catch(IOException e) {
    e.printStackTrace();
}
}

private void downloadJsonFile() {
try{
    createFileAndDirectory();
    URL url = new URL(UsersController.DL_URL);
    HttpURLConnection urlConnection;
    urlConnection = (HttpURLConnection) url.openConnection();
    urlConnection.setRequestMethod("GET");
    urlConnection.setDoOutput(true);
    urlConnection.connect();
    FileOutputStream fileOutput = new FileOutputStream(jsonFile);
    InputStream inputStream = urlConnection.getInputStream();
    byte[] buffer = newbyte[1024];
    int bufferLength = 0;
    while((bufferLength = inputStream.read(buffer)) > 0) {
    fileOutput.write(buffer, 0, bufferLength);
    }
    fileOutput.close();
} catch(MalformedURLException e) {
    e.printStackTrace();
} catch(IOException e) {
    e.printStackTrace();
}
}

private void createFileAndDirectory() throws FileNotFoundException {
final String extStorageDirectory = Environment
    .getExternalStorageDirectory().toString();
final String meteoDirectory_path = extStorageDirectory + "/tutos-android";
jsonOutputFile = new File(meteoDirectory_path, "/");
if(jsonOutputFile.exists() == false)
    jsonOutputFile.mkdirs();
jsonFile = new File(jsonOutputFile, "users.json");
}

public ArrayList<User> findAll() {
return userList;
}

public User findById(intid) {
return userList.get(id);
}
```

```
}
```

Cette classe s'occupera de :

- créer un dossier “**tutos-android** sur le téléphone, puis un fichier **users.json** (**createFileAndDirectory**) ;
- ouvrir une connexion avec le serveur pour lire le contenu du **JSON** et le recopier localement (**downloadJsonFile**) ;
- initialiser notre parseur, à l'aide du fichier JSON local :

```
jp = jsonFactory.createJsonParser(jsonFile);
```

- Lire des valeurs portant la balise “**Users** :

```
users = objectMapper.readValue(jp, Users.class);
```

- Récupérer de la liste des utilisateurs :

```
userList = users.get("Users");
```

Comme vous venez de le remarquer, parser un fichier JSON est très simple et rapide.

Pour finir, nous allons implémenter notre activité pour :

- télécharger notre fichier JSON ;
- afficher le résultat dans le TextView :

```
package com.tutos.android.json;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

import com.tutos.android.json.model.User;
import com.tutos.android.json.model.UsersController;

public class JsonParserMainActivity extendsActivity {
    private UsersController usersController;
    private TextView displayJson;

    @Override
    public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    usersController = newUsersController();

    displayJson = (TextView) findViewById(R.id.jsonDisplay);

    Button startParsing = (Button) findViewById(R.id.startParsing);
    startParsing.setOnClickListener(newOnClickListener() {
        @Override
        public void onClick(View view) {
        gettingJson();
        }
    });
    }

    final void gettingJson() {
    final Thread checkUpdate = new Thread() {
        public void run() {
        usersController.init();
        final StringBuilder str = newStringBuilder("user : ");
```

```
        for(User u : usersController.findAll()) {
            str.append("\n").append("first name : ").append(u.getFirstname());
            str.append("\n").append("last name : ").append(u.getLastname());
            str.append("\n").append("login : ").append(u.getLogin());
            str.append("\n").append("twitter : ").append(u.getTwitter());
            str.append("\n").append("Web : ").append(u.getWeb());
        }
        runOnUiThread(newRunnable() {
                @Override
                public void run() {
                    displayJson.setText(str.toString());
                }
            });

        }
    };
    checkUpdate.start();
    }

}
```

Ce qui vous donnera le résultat suivant :

Pour finir je vous fournis le .zip contenant le projet, vous pouvez le télécharger **ici**.

4 - Conclusion

Ce tutoriel s'arrête ici en espérant qu'il vous a aidé a comprendre comment fonctionne le parsing JSON sous Android, l'utilité de la bibliothèque Jackson ainsi que l'avantage que peux procurer JSON sur d'autres formats disponibles comme XML par exemple.

5 - Remerciements

Je tiens à remercier tout particulièrement **djibril** qui a mis ce tutoriel au format Developpez.com.
Merci également à **jacques_jean** d'avoir pris le temps de le relire et de le corriger.

6 - Liens

Tutoriel origine sur Tutos-Android