Atitit.json类库的设计与实现 ati json lib

[1. 目前jsonlib库可能有问题，可能版本冲突，抛出ex 1](#_Toc10893)

[2. 解决之道: 1](#_Toc13512)

[2.1. 自定义json解析库，使用多个复合的json 解析复合的引擎 1](#_Toc17647)

[3. 几个要点 2](#_Toc11233)

[3.1. 复合结构的转换 2](#_Toc1286)

[3.2. 没有get set方法的属性自动忽略，而不抛出异常 2](#_Toc4888)

[3.3. 时间格式的转换 2](#_Toc4621)

[3.4. Api使用json lib的,以及gson的也一个 2](#_Toc30877)

[3.5. 如果要将泛型转换成json， 2](#_Toc15825)

[4. 普通的的json解析器 Jackson类库 Google Gson JSON-lib类库 3](#_Toc23037)

[5. Api 3](#_Toc32465)

[5.1. toJson xxx.fromObject(x).toString(2) 3](#_Toc11833)

[5.2. 字符串转成对象  T fromJson(String str, Class<T> type) fromObject(object) 3](#_Toc26761)

[6. Teste code 3](#_Toc14706)

# 目前jsonlib库可能有问题，可能版本冲突，抛出ex

# 解决之道:

## 自定义json解析库，使用多个复合的json 解析复合的引擎

# 几个要点

## 复合结构的转换

## 没有get set方法的属性自动忽略，而不抛出异常

## 时间格式的转换

## Api使用json lib的,以及gson的也一个

作者:: 老哇的爪子 Attilax 艾龙，  EMAIL:1466519819@qq.com

转载请注明来源： http://blog.csdn.net/attilax

## 如果要将泛型转换成json，

1. 如，一个Map是这样的，Map<String, Map<String, List<String>>> map，如果这样使用gson.toJson(map)得不到正确的结果的话，那么，我们可能要这样使用，gson.toJson(map,new TypeToken<Map<String, Map<String, List<String>>>>(){}.getType())。

**解决方案：**

　　1、如果涉及到关联属性的对象，转换前可以先把它的关联属性转换并放入一个新建的Map或者List，然后按照层次结构重复这样的操作一层一层的往上转，这样，一般可以解决问题，但是，如果关联的层次比较深，做起来就很麻烦了。上面提到的问题3就可以用这种方式解决。

# 普通的的json解析器 ****Jackson类库 Google**** Gson ****JSON-lib类库****

# ****Api****

## ****toJson xxx.fromObject(x).toString(2)****

## 字符串转成对象  T fromJson(String str, Class<T> type) fromObject(object)

# ****Teste code****

**package** com.attilax.json;

**import** java.util.Date;

**import** java.util.HashMap;

**import** java.util.Map;

**import** com.attilax.core;

**import** com.google.gson.Gson;

**public** **class** JSONObject {

**public** JSONObject(Map m) {

**this**.obj = m;

}

**public** **static** **void** main(String[] args) {

Map m = **new** HashMap();

m.put("d", **new** Date());

// m.put(key, value)

JSONObject.*fromObject*(m).toString(2);

}

**private** String toString(**int** i) {

// if(i==2) //fmt

{

**try** {

**return** net.sf.json.JSONObject.*fromObject*(**this**.obj).toString(i);

} **catch** (Exception e) {

**try** {

// 创建一个gson对象

Gson gson = **new** Gson();

// 转换成json

String json = gson.toJson(**this**.obj);

**return** json;

} **catch** (Exception e2) {

**return** JsonUtil4jackjson.*buildNormalBinder*().toJson(

**this**.obj);

}

}

}

}

**public** Object obj;

**private** **static** JSONObject fromObject(Map m) {

// **TODO** Auto-generated method stub

**return** **new** JSONObject(m);

}

}