## 一、背景

21世纪初，Ddouglas Crockford寻找一种简便的数据交换格式，能够在服务器之间交换数据，  
当时通过数据交换语言是XML,但是Douglas Crockford觉得XML的生成和解析都太麻烦了，所以  
提出了一种简化的数据格式JSON  
JSON的规格非常简单，只用一个页面几百个字就能说的清除，而且Douglas Crockford声称这个

规格永远不必升级，应为该规定的都规定了

## 二、三种基本数据格式类型

第一种是标量（scalar）,也就是单个字符串（Stirng）或是数字（number）,如“上海”这个单独的词  
  
第二种类型是序列（sequence）,也就是相关的数据按照一定的顺序并列在一起，又叫做数组（array）

或列表（list）,如“北京,上海”.

第三种类型是映射（mapping）,也就是键/值对（key/value）,即数据有一个名称，还有一个与之对应  
的值，又称作散列（hash）或字典（dictionnary）,比如“首都：北京”。

## 三、什么是JSON

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。  
它基于ECMAScript的一个子集。 JSON采用完全独立于语言的文本格式，  
但是也使用了类似于C语言家族的习惯

#### 优点：

1.支持多语言包括C、C++、C#、Java、JavaScript、Perl、Python等）。这些特性使JSON成为理想的数据交换语言。  
2.易于人阅读和编写，同时也易于机器解析和生成(一般用于提升网络传输速率)。

#### 缺点：

1.要求字符集必须是Unicode,约束性强  
2.语法严禁，必须符合JSON语法规则  
  
  
案例：  
{"name":"tom","age":20}  
{"first":{"name":"tom","age":20},"second":{"name":"jack","age":24}}  
[{'name':'成龙','age':50},{'name':'李冰冰','age':35}]

## 四、四个基本原则

1.并列的数据之间使用（“,”）f分别  
2.映射用冒号（“:”）标识  
3.并列数据的集合（数组）使用方括号（“[]”）标识

4.映射的集合（对象）用大括号（“{}”）标识

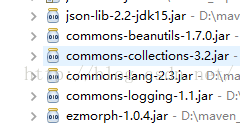
## 五、JSON在JAVA中的使用

json在**[Java](http://lib.csdn.net/base/javase" \o "Java SE知识库" \t "/home/caojx/文档\\x/_blank)**中的使用其实就是编码和解码的过程

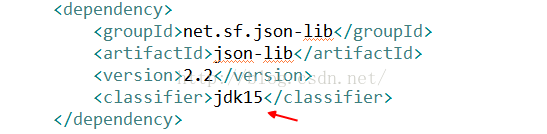
编码：将数据编码成json格式

解码：将json个数是数据或对象转为你需要的类型

使用json需要的jar包



如果是maven工程，如下配置可以引入json的系列jar包



注：<classifier>jdk15</classifier>这里是要求写上的，声明jdk版本实现，不然无法引入，因为还有jdk13和jdk15两个版本实现之分

两个基本对象JSONObject和JSONArray于其他数据进行转换

#### 5.1JSONObject

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">    </span>JSONObject json = **new** JSONObject();
2. json.put("name", "tom");
3. json.put("age", 20);
4. System.out.println(json);//{"age":20,"name":"tom"}
5. /\*
6. \* 下边的取值方式，第一种get通过key取值，返回Object类型的value值
7. \*              第二种getString取出指定类型的value值，如下返回String类型，取不到值会抛出异常
8. \*              第三种optString取出指定类型的value值，取不到值不会抛出异常
9. \* \*/
10. System.out.println(json.get("name"));
11. System.out.println(json.getString("name"));
12. System.out.println(json.optString("name"));

#### 5.2JSONArray

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">        </span>/\*
2. \* JSONArray
3. \* 存放数据的时候需要逐个存放，从0索引开始，
4. \* 不能越界存放，需要将前边的存放后才能存后边的
5. \* 如：不能索引存了1后边的索引为3，只能一次递增存放，不让会报索引越界异常
6. \* 存放的时候，索引可以不写默认从0递增
7. \* 取值的时候，可以通过索引来取
8. \*/
9. JSONArray jsonArray = **new** JSONArray();
10. jsonArray.add(0,"张三");
11. jsonArray.add(1,"李四");
12. jsonArray.add(2,"王五");
13. jsonArray.add(3,"赵六");
14. System.out.println(jsonArray);//["张三","李四","王五","赵六"]
15. System.out.println(jsonArray.get(0));//张三

#### 5.3 将多个JSONObject--->JSONArray

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">        </span>JSONObject json1 = **new** JSONObject();
2. json1.put("name", "rose");
3. json1.put("age", 20);
4. JSONObject json2 = **new** JSONObject();
5. json2.put("name", "jack");
6. json2.put("age", 20);
8. JSONArray jsonArray2 = **new** JSONArray();
9. jsonArray2.add(json1);
10. jsonArray2.add(json2);
11. System.out.println(jsonArray2);//[{"age":20,"name":"rose"},{"age":20,"name":"jack"}]
12. System.out.println(jsonArray2.get(1));//{"age":20,"name":"jack"}

#### 5.4JSONArray ---> JSONObject

通过从JSONArray通过索引取出JSONObject

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">        </span>JSONObject jsonObject1 = jsonArray2.getJSONObject(0);
2. System.out.println("JSONArray ---> JSONObject:"+jsonObject1);

#### 5.5(JSON)String --> JSONObject

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">    </span>String jsonStr = "{'name':'成龙','age':50}";
2. JSONObject json3 = JSONObject.fromObject(jsonStr);
3. System.out.println(json3);//{"age":50,"name":"成龙"}

#### 5.6(JSON)String ---> JSONArray

json字符串需要转成JSONArray必须是json字符串数组

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. String jsonStrs = "[{'name':'成龙','age':50},{'name':'李冰冰','age':35}]";
2. JSONArray jsonArray4 = JSONArray.fromObject(jsonStrs);
3. System.out.println(jsonArray4);//[{"age":50,"name":"成龙"},{"age":35,"name":"李冰冰"}]

#### 5.7Object ---> JSONObejct or JSONArray

User(bean) --> JSONObject

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">    </span>/\*
2. \* User(bean) --> JSONObject
3. \* 会将bean的属性作为key,值作为value
4. \*/
5. User user = **new** User();
6. user.setName("tom");
7. user.setAge(20);
8. JSONObject jsonObject2 = JSONObject.fromObject(user);
9. System.err.println(jsonObject2);//{"name":"tom","age":20}

list --->JSONArray

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">        </span>User user2 = **new** User();
2. user2.setName("jack");
3. user2.setAge(21);
4. List<User> list=**new** ArrayList<User>();
5. list.add(user);
6. list.add(user2);
7. JSONArray jsonArray3 = JSONArray.fromObject(list);
8. System.out.println(jsonArray3);//[{"age":20,"name":"tom"},{"age":21,"name":"jack"}]

#### 5.8 jsonStirng--->Bean

(JSON)String -->JSONObeject---Bean  
key值对应bean中的属性值  
json字符串 中key可以不加""或'',即可以为任意类型,不过value值如果为字符串一定要加""或'',不是字符串可以不加  
不过建议key都使用""或''

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">    </span>String jsonStr2 = "{'name':'成龙','age':50}";
2. JSONObject json4 = JSONObject.fromObject(jsonStr2);
3. User user3 = (User) JSONObject.toBean(json4, User.**class**);
4. System.out.println(user3.toString());//User [name=成龙, age=50]

(JSON)String --->JSONArray-->Bean[]

**[java]** [view plain](http://blog.csdn.net/u014292162/article/details/52191232" \o "view plain) [copy](http://blog.csdn.net/u014292162/article/details/52191232" \o "copy)

1. <span style="white-space:pre">    </span>String jsonStrs2 = "[{'name':'成龙','age':50},{'name':'李冰冰','age':35}]";
2. JSONArray jsonArray5 = JSONArray.fromObject(jsonStrs2);
3. User[] userS = (User[]) JSONArray.toArray(jsonArray5, User.**class**);
4. System.out.println(userS[0]);//User [name=成龙, age=50]

#### 5.9JSON-->MAP or List

http://www.tuicool.com/articles/BzQrayA