[**JSON 数据格式**](http://www.cnblogs.com/SkySoot/archive/2012/04/17/2453010.html)

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。JSON采用完全独立于语言的文本格式，这些特性使JSON成为理想的数据交换语言。易于人阅读和编写，同时也易于机器解析和生成。

**基础结构**

JSON建构于两种结构：

1. “名称/值”对的集合（A collection of name/value pairs）。不同的语言中，它被理解为对象（object），记录（record），结构（struct），字典（dictionary），哈希表（hash table），有键列表（keyed list），或者关联数组 （associative array）。

2. 值的有序列表（An ordered list of values）。在大部分语言中，它被理解为数组（array）。

**基础示例**

简单地说，JSON 可以将 JavaScript 对象中表示的一组数据转换为字符串，然后就可以在函数之间轻松地传递这个字符串，或者在异步应用程序中将字符串从 Web 客户机传递给服务器端程序。这个字符串看起来有点儿古怪，但是 JavaScript 很容易解释它，而且 JSON 可以表示比"名称 / 值对"更复杂的结构。例如，可以表示数组和复杂的对象，而不仅仅是键和值的简单列表。

**表示名称 / 值对**

按照最简单的形式，可以用下面这样的 JSON 表示 "名称 / 值对" ：{ "firstName": "Brett" }

这个示例非常基本，而且实际上比等效的纯文本 "名称 / 值对" 占用更多的空间：firstName=Brett

但是，当将多个"名称 / 值对"串在一起时，JSON 就会体现出它的价值了。首先，可以创建包含多个"名称 / 值对"的 记录，比如：

{ "firstName": "Brett", "lastName":"McLaughlin", "email": "aaaa" }

从语法方面来看，这与"名称 / 值对"相比并没有很大的优势，但是在这种情况下 JSON 更容易使用，而且可读性更好。例如，它明确地表示以上三个值都是同一记录的一部分；花括号使这些值有了某种联系。

**表示数组**

当需要表示一组值时，JSON 不但能够提高可读性，而且可以减少复杂性。例如，假设您希望表示一个人名列表。在 XML 中，需要许多开始标记和结束标记；如果使用典型的 名称 / 值 对（就像在本系列前面文章中看到的那种名称 / 值对），那么必须建立一种专有的数据格式，或者将键名称修改为 person1-firstName这样的形式。

如果使用 JSON，就只需将多个带花括号的记录分组在一起：

{ "people": [

{ "firstName": "Brett", "lastName":"McLaughlin", "email": "aaaa" },

{ "firstName": "Jason", "lastName":"Hunter", "email": "bbbb"},

{ "firstName": "Elliotte", "lastName":"Harold", "email": "cccc" }

]}

这不难理解。在这个示例中，只有一个名为 people的变量，值是包含三个条目的数组，每个条目是一个人的记录，其中包含名、姓和电子邮件地址。上面的示例演示如何用括号将记录组合成一个值。当然，可以使用相同的语法表示多个值（每个值包含多个记录）：

{ "programmers": [

{ "firstName": "Brett", "lastName":"McLaughlin", "email": "aaaa" },

{ "firstName": "Jason", "lastName":"Hunter", "email": "bbbb" },

{ "firstName": "Elliotte", "lastName":"Harold", "email": "cccc" }

],

"authors": [

{ "firstName": "Isaac", "lastName": "Asimov", "genre": "science fiction" },

{ "firstName": "Tad", "lastName": "Williams", "genre": "fantasy" },

{ "firstName": "Frank", "lastName": "Peretti", "genre": "christian fiction" }

],

"musicians": [

{ "firstName": "Eric", "lastName": "Clapton", "instrument": "guitar" },

{ "firstName": "Sergei", "lastName": "Rachmaninoff", "instrument": "piano" }

] }

这里最值得注意的是，能够表示多个值，每个值进而包含多个值。但是还应该注意，在不同的主条目（programmers、authors 和 musicians）之间，记录中实际的名称 / 值对可以不一样。JSON 是完全动态的，允许在 JSON 结构的中间改变表示数据的方式。

在处理 JSON 格式的数据时，没有需要遵守的预定义的约束。所以，在同样的数据结构中，可以改变表示数据的方式，甚至可以以不同方式表示同一事物。

**格式应用**

掌握了 JSON 格式之后，在 JavaScript 中使用它就很简单了。JSON 是 JavaScript 原生格式，这意味着在 JavaScript 中处理 JSON 数据不需要任何特殊的 API 或工具包。

**将 JSON 数据赋值给变量**

例如，可以创建一个新的 JavaScript 变量，然后将 JSON 格式的数据字符串直接赋值给它：

var people = { "programmers": [ { "firstName": "Brett", "lastName":"McLaughlin", "email": "aaaa" },

{ "firstName": "Jason", "lastName":"Hunter", "email": "bbbb" },

{ "firstName": "Elliotte", "lastName":"Harold", "email": "cccc" }

],

"authors": [

{ "firstName": "Isaac", "lastName": "Asimov", "genre": "science fiction" },

{ "firstName": "Tad", "lastName": "Williams", "genre": "fantasy" },

{ "firstName": "Frank", "lastName": "Peretti", "genre": "christian fiction" }

],

"musicians": [

{ "firstName": "Eric", "lastName": "Clapton", "instrument": "guitar" },

{ "firstName": "Sergei", "lastName": "Rachmaninoff", "instrument": "piano" }

] }

这非常简单；现在 people包含前面看到的 JSON 格式的数据。但是，这还不够，因为访问数据的方式似乎还不明显。

**访问数据**

尽管看起来不明显，但是上面的长字符串实际上只是一个数组；将这个数组放进 JavaScript 变量之后，就可以很轻松地访问它。实际上，只需用点号表示法来表示数组元素。所以，要想访问 programmers 列表的第一个条目的姓氏，只需在 JavaScript 中使用下面这样的代码：

people.programmers[0].lastName;

注意，数组索引是从零开始的。所以，这行代码首先访问 people变量中的数据；然后移动到称为 programmers的条目，再移动到第一个记录（[0]）；最后，访问 lastName键的值。结果是字符串值 “McLaughlin”。

下面是使用同一变量的几个示例。

people.authors[1].genre // Value is "fantasy"

people.musicians[3].lastName // Undefined. This refers to the fourth entry, and there isn't one

people.programmers[2].firstName // Value is "Elliotte"

利用这样的语法，可以处理任何 JSON 格式的数据，而不需要使用任何额外的 JavaScript 工具包或 API。

**修改 JSON 数据**

正如可以用点号和括号访问数据，也可以按照同样的方式轻松地修改数据：

people.musicians[1].lastName = "Rachmaninov";

在将字符串转换为 JavaScript 对象之后，就可以像这样修改变量中的数据。

**转换回字符串**

当然，如果不能轻松地将对象转换回本文提到的文本格式，那么所有数据修改都没有太大的价值。在 JavaScript 中这种转换也很简单：

String newJSONtext = people.toJSONString();

这样就行了！现在就获得了一个可以在任何地方使用的文本字符串，例如，可以将它用作 Ajax 应用程序中的请求字符串。

更重要的是，可以将任何JavaScript 对象转换为 JSON 文本。并非只能处理原来用 JSON 字符串赋值的变量。为了对名为 myObject的对象进行转换，只需执行相同形式的命令：

String myObjectInJSON = myObject.toJSONString();

这就是 JSON 与本系列讨论的其他数据格式之间最大的差异。如果使用 JSON，只需调用一个简单的函数，就可以获得经过格式化的数据，可以直接使用了。对于其他数据格式，需要在原始数据和格式化数据之间进行转换。即使使用 Document Object Model 这样的 API（提供了将自己的数据结构转换为文本的函数），也需要学习这个 API 并使用 API 的对象，而不是使用原生的 JavaScript 对象和语法。

最终结论是，如果要处理大量 JavaScript 对象，那么 JSON 几乎肯定是一个好选择，这样就可以轻松地将数据转换为可以在请求中发送给服务器端程序的格式。

**概念比较**

**JSON和XML的比较**

◆**可读性**

JSON和XML的可读性可谓不相上下，一边是简易的语法，一边是规范的标签形式，很难分出胜负。

◆**可扩展性**

XML天生有很好的扩展性，JSON当然也有，没有什么是XML能扩展，而JSON却不能。不过JSON在Javascript主场作战，可以存储Javascript复合对象，有着xml不可比拟的优势。

◆**编码难度**

XML有丰富的编码工具，比如Dom4j、JDom等，JSON也有提供的工具。无工具的情况下，相信熟练的开发人员一样能很快的写出想要的xml文档和JSON字符串，不过，xml文档要多很多结构上的字符。

◆**解码难度**

XML的解析方式有两种：

一是通过文档模型解析，也就是通过父标签索引出一组标记。例如：xmlData.getElementsByTagName("tagName")，但是这样是要在预先知道文档结构的情况下使用，无法进行通用的封装。

另外一种方法是遍历节点（document 以及 childNodes）。这个可以通过递归来实现，不过解析出来的数据仍旧是形式各异，往往也不能满足预先的要求。

凡是这样可扩展的结构数据解析起来一定都很困难。

JSON也同样如此。如果预先知道JSON结构的情况下，使用JSON进行数据传递简直是太美妙了，可以写出很实用美观可读性强的代码。如果你是纯粹的前台开发人员，一定会非常喜欢JSON。但是如果你是一个应用开发人员，就不是那么喜欢了，毕竟xml才是真正的结构化标记语言，用于进行数据传递。

而如果不知道JSON的结构而去解析JSON的话，那简直是噩梦。费时费力不说，代码也会变得冗余拖沓，得到的结果也不尽人意。但是这样也不影响众多前台开发人员选择JSON。因为json.js中的toJSONString()就可以看到JSON的字符串结构。当然不是使用这个字符串，这样仍旧是噩梦。常用JSON的人看到这个字符串之后，就对JSON的结构很明了了，就更容易的操作JSON。

以上是在Javascript中仅对于数据传递的xml与JSON的解析。在Javascript地盘内，JSON毕竟是主场作战，其优势当然要远远优越于xml。如果JSON中存储Javascript复合对象，而且不知道其结构的话，我相信很多程序员也一样是哭着解析JSON的。

◆**实例比较**

XML和JSON都使用结构化方法来标记数据，下面来做一个简单的比较。

用XML表示[中国](http://baike.baidu.com/view/61891.htm)部分省市数据如下：

<?xml version="1.0" encoding="utf-8"?>

<country>

<name>中国</name>

<province>

<name>黑龙江</name>

<cities>

<city>哈尔滨</city>

<city>大庆</city>

</cities>

</province>

<province>

<name>广东</name>

<cities>

<city>广州</city>

<city>深圳</city>

<city>珠海</city>

</cities>

</province>

</country>

用JSON表示如下：

{

{name:"中国", province:[ { name:"黑龙江", cities:{ city:["哈尔滨","大庆"] },

{name:"广东", cities:{ city:["广州","深圳","珠海"] }

}

编码的可读性，xml有明显的优势，毕竟人类的语言更贴近这样的说明结构。json读起来更像一个数据块，读起来就比较费解了。不过，我们读起来费解的语言，恰恰是适合机器阅读，所以通过json的索引.province[0].name就能够读取“黑龙江”这个值。

编码的手写难度来说，xml还是舒服一些，好读当然就好写。不过写出来的字符JSON就明显少很多。去掉空白制表以及换行的话，JSON就是密密麻麻的有用数据，而xml却包含很多重复的标记字符。

**JSON在线校验工具**

**前言**

JSON格式取代了xml给网络传输带来了很大的便利,但是却没有了xml的一目了然,尤其是json数据很长的时候,我们会陷入繁琐复杂的数据节点查找中。

但是国人的一款在线工具 BeJson 给众多程序员带来了一阵凉风。

**功能简介**

**1. JSON格式化校验**

很多人在得到JSON数据后,一时没有办法判断JSON数据格式是否正确,是否少或多符号而导致程序不能解析,这个功能正好能帮助大家来完成JSON格式的校验。

**2. JSON视图**

想必很多程序员都会遇到当找一个节点的时候,会发现如果直接对着一行行数据无从下手,就算知道哪个位置,还要一个节点一个节点的往下找,万一一不留神又得从头开始找的麻烦事。

有了这个功能,一切JSON数据都会变成视图格式,一目了然,什么对象下有多少数组,一个数组下有多少对象。

这个功能非常实用。不光有视图功能还有格式化、压缩、转义、校验功能。总之很强大。

**3. 压缩转义**

程序员在写JSON语句测试用例的时候,很多时候为了方便直接写了个JSON字符串做测试,但是又陷入了无止境的双引号转义的麻烦中。这款功能集压缩、转义于一身,让你在写测试用例的时候,如鱼得水。

**4. JSON在线编辑器**

如果你现在的电脑刚巧没有装你所熟悉的编辑器,如果你想针对拿到的JSON数据的某个节点做数据修改时,这个功能可以满足你的需求。

**5. 在线发送JSON数据**

大家都知道,JSON用的最多的还是web项目的开发,那你要测试一个接口是否能准确的接受JSON数据,那你就得写一个页面发送JSON字符串,重复的做着这件事。随着这个功能的横空出世,你可以摆脱写测试页面了,因为这个功能可以将指定的JSON数据发送指定的url,方便吧。

**6. JSON着色**

很多人在写文档时，总希望文档能一目了然,但是面对着白底黑字的JSON数据总是提不起精神没关系,使用这个功能,所有的关键字都会被着色,数据结构一目了然。

**7. JSON-XML互转**

顾名思义，将JSON格式的数据转化成XML格式、或者XML格式的数据转化成JSON格式，一切都不是问题。

1.什么是JSON  
JSON是一种轻量级的数据交换格式，全称是：JavaScript Object Notation，它有完全独立的数据格式，所以它是独立于任何编程语言的，它不太易于阅读，但是易于机器解析和生成。  
2.JSON数据格式  
JSON有两种格式  
  
  
名称/值 成对的集合，类似于关联数组或者map结构。  
实例：  
{"name": "eliteYang"}  
这个结构显然没有name=eliteYang来的直接和明显，而且浪费了很多引号，但是如果多个结构就会体现出这种结构的优势了，例如：  
{"name": "eliteYang", "name": "AAA", "name"="BBB"}  
前面我说过，这种结构不太容易阅读，这里它明显看起来很纠结，所以有些人说这个容易阅读让我很费解，大概是写代码写多了，接近机器了吧。  
  
  
值的有序列表，大部分语言中，这种结构被称之为数组。  
实例：  
{"address":[  
    {"name":"eliteYang", "email":"elite\_yang@163.com"},  
    {"name":"AAA", "email":"aaa@163.com"},  
    {"name":"BBB", "email":"bbb@163.com"}  
]}  
  
  
  
  
  
3.JSON的用途  
JSON主要用于数据传输和转化，常用在web服务中，例如本站用的多说评论插件，它就是使用JSON将评论数据同步到多说的服务器上的，同时，显示评论也是从多说的服务器上通过JSON请求过来的。JS非常容易操作JSON，因为JSON的数据格式是JS原生的数据格式。当然了，其他方面也可以使用，比如网络传输，将JSON转化成字符串，压缩加密后用于消息事件，或者存储数据库时，将一些数据整个存成JSON格式，然后做成一个字符串存入数据库，还有许多用途，这里就不一一介绍了。  
  
PS:  
JSON与XML格式的比较：  
优点：JSON占用较少字符，主要用在JS上，对存储JS复合对象来说很方便  
缺点：JSON不易于阅读，可扩展性不强，而XML天生就有很强的扩展性，并且完全符合人类的阅读习惯。  
而从编写角度来说，都差不多，因为现在都有专门的工具来编写两种格式的文件，解码难度也是近似的，都需要知道节点名，然后读取出数据。