[**JSON详解**](http://www.cnblogs.com/mcgrady/archive/2013/06/08/3127781.html)

JSON的全称是”JavaScript Object Notation”，意思是JavaScript对象表示法，它是一种基于文本，独立于语言的轻量级数据交换格式。XML也是一种数据交换格式，为什么没有选择XML呢？因为XML虽然可以作为跨平台的数据交换格式，但是在JS(JavaScript的简写)中处理XML非常不方便，同时XML标记比数据多，增加了交换产生的流量，而JSON没有附加的任何标记，在JS中可作为对象处理，所以我们更倾向于选择JSON来交换数据。这篇文章主要从以下几个方面来说明JSON。

1，JSON的两种结构  
2，认识JSON字符串  
3，在JS中如何使用JSON  
4，在.NET中如何使用JSON  
5，总结

**JSON的两种结构**

JSON有两种表示结构，对象和数组。  
对象结构以”{”大括号开始，以”}”大括号结束。中间部分由0或多个以”，”分隔的”key(关键字)/value(值)”对构成，关键字和值之间以”：”分隔，语法结构如代码。

{

key1:value1,

key2:value2,

...

}

其中关键字是字符串，而值可以是字符串，数值，true,false,null,对象或数组

数组结构以”[”开始，”]”结束。中间由0或多个以”，”分隔的值列表组成，语法结构如代码。

[复制代码](javascript:void(0);)

[

{

key1:value1,

key2:value2

},

{

key3:value3,

key4:value4

}

]

[复制代码](javascript:void(0);)

**认识JSON字符串**

之前我一直有个困惑，分不清普通字符串，json字符串和json对象的区别。经过一番研究终于给弄明白了。比如在js中。

字符串：这个很好解释，指使用“”双引号或’’单引号包括的字符。例如：var comStr = 'this is string';  
json字符串：指的是符合json格式要求的js字符串。例如：var jsonStr = "{StudentID:'100',Name:'tmac',Hometown:'usa'}";  
json对象：指符合json格式要求的js对象。例如：var jsonObj = { StudentID: "100", Name: "tmac", Hometown: "usa" };

**在JS中如何使用JSON**

JSON是JS的一个子集，所以可以在JS中轻松地读，写JSON。读和写JSON都有两种方法，分别是利用”.”操作符和“[key]”的方式。  
我们首先定义一个JSON对象，代码如下。

[复制代码](javascript:void(0);)

var obj = {

1: "value1",

"2": "value2",

count: 3,

person: [ //数组结构JSON对象，可以嵌套使用

{

id: 1,

name: "张三"

},

{

id: 2,

name: "李四"

}

],

object: { //对象结构JSON对象

id: 1,

msg: "对象里的对象"

}

};

[复制代码](javascript:void(0);)

1，从JSON中读数据

[复制代码](javascript:void(0);)

function ReadJSON() {

alert(obj.1); //会报语法错误，可以用alert(obj["1"]);说明数字最好不要做关键字

alert(obj.2); //同上

alert(obj.person[0].name); //或者alert(obj.person[0]["name"])

alert(obj.object.msg); //或者alert(obj.object["msg"])

}

[复制代码](javascript:void(0);)

2，向JSON中写数据

比如要往JSON中增加一条数据，代码如下：

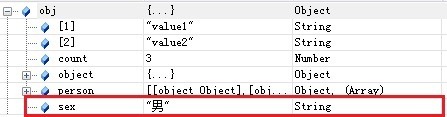
function Add() {

//往JSON对象中增加了一条记录

obj.sex= "男" //或者obj["sex"]="男"

}

增加数据后的JSON对象如图：

[](http://images.cnitblog.com/blog/311549/201306/08225647-1f3c6b7829074c529ee3365315ccc6b0.jpg)

3，修改JSON中的数据

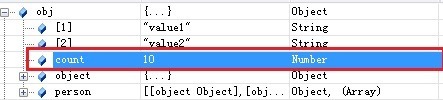
我们现在要修改JSON中count的值，代码如下：

function Update() {

obj.count = 10; //或obj["count"]=10

}

修改后的JSON如图。

[](http://images.cnitblog.com/blog/311549/201306/08225654-7373ca8222ad4b999701ce3f988d5d68.jpg)

4，删除JSON中的数据

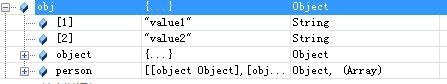
我们现在实现从JSON中删除count这条数据，代码如下：

function Delete() {

delete obj.count;

}

删除后的JSON如图

[](http://images.cnitblog.com/blog/311549/201306/08225700-1fc21b4018054d0b8d23477a035eb15f.jpg)

可以看到count已经从JSON对象中被删除了。

5，遍历JSON对象

可以使用for…in…循环来遍历JSON对象中的数据，比如我们要遍历输出obj对象的值，代码如下：

function Traversal() {

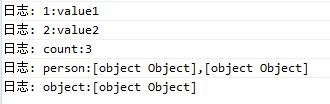
for (var c in obj) {

console.log(c + ":", obj[c]);

}

}

程序输出结果为：

[](http://images.cnitblog.com/blog/311549/201306/08225705-188a9dd782ca4f42934fcba606e2b162.jpg)

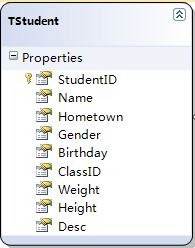
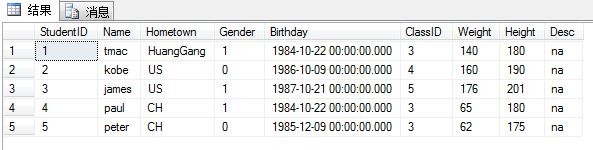
**在.NET中如何使用JSON**

说到在.net中使用JSON，就不得不提到JSON.NET，它是一个非常著名的在.net中处理JSON的工具，我们最常用的是下面两个功能。

**1，通过序列化将.net对象转换为JSON字符串**

在web开发过程中，我们经常需要将从数据库中查询到的数据(一般为一个集合，列表或数组等)转换为JSON格式字符串传回客户端，这就需要进行序列化，这里用到的是JsonConvert对象的SerializeObject方法。其语法格式为：JsonConvert.SerializeObject(object)，代码中的”object”就是要序列化的.net对象，序列化后返回的是json字符串。

比如，现在我们有一个TStudent的学生表，表中的字段和已有数据如图所示

[](http://images.cnitblog.com/blog/311549/201306/10165638-39a0521daa1a49b5b94ee8b764699ce4.jpg)[](http://images.cnitblog.com/blog/311549/201306/10165646-be2e8e7ef4454ba0a12befa763b90b16.jpg)

从表中我们可以看到一共有五条数据，现在我们要从数据库中取出这些数据，然后利用JSON.NET的JsonConvert对象序列化它们为json字符串，并显示在页面上。C#代码如下

[复制代码](javascript:void(0);)

protected void Page\_Load(object sender, EventArgs e)

{

using (L2SDBDataContext db = new L2SDBDataContext())

{

List<Student> studentList = new List<Student>();

var query = from s in db.TStudents

select new {

StudentID=s.StudentID,

Name=s.Name,

Hometown=s.Hometown,

Gender=s.Gender,

Brithday=s.Birthday,

ClassID=s.ClassID,

Weight=s.Weight,

Height=s.Height,

Desc=s.Desc

};

foreach (var item in query)

{

Student student = new Student { StudentID=item.StudentID,Name=item.Name,Hometown=item.Hometown,Gender=item.Gender,Brithday=item.Brithday,ClassID=item.ClassID,Weight=item.Weight,Height=item.Height,Desc=item.Desc};

studentList.Add(student);

}

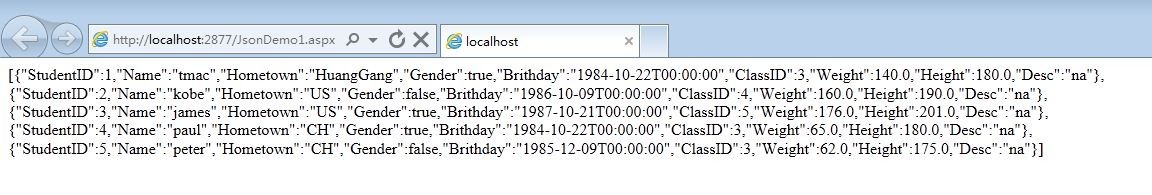
lbMsg.InnerText = JsonConvert.SerializeObject(studentList);

}

}

[复制代码](javascript:void(0);)

输出结果

[](http://images.cnitblog.com/blog/311549/201306/10165700-09eab47cea0d4d61a62e93d63937a40c.jpg)

从图中我们可以看到，数据库中的5条记录全部取出来并转化为json字符串了。

**2，使用LINQ to JSON定制JSON数据**

使用JsonConvert对象的SerializeObject只是简单地将一个list或集合转换为json字符串。但是，有的时候我们的前端框架比如ExtJs对服务端返回的数据格式是有一定要求的，比如下面的数据格式，这时就需要用到JSON.NET的LINQ to JSON，LINQ to JSON的作用就是根据需要的格式来定制json数据。

比如经常用在分页的json格式如代码：

{

"total": 5, //记录总数

"rows":[

//json格式的数据列表

]

}

使用LINQ to JSON前，需要引用Newtonsoft.Json的dll和using Newtonsoft.Json.Linq的命名空间。LINQ to JSON主要使用到JObject, JArray, JProperty和JValue这四个对象，JObject用来生成一个JSON对象，简单来说就是生成”{}”，JArray用来生成一个JSON数组，也就是”[]”，JProperty用来生成一个JSON数据，格式为key/value的值，而JValue则直接生成一个JSON值。下面我们就用LINQ to JSON返回上面分页格式的数据。代码如下：

[复制代码](javascript:void(0);)

protected void Page\_Load(object sender, EventArgs e)

{

using (L2SDBDataContext db = new L2SDBDataContext())

{

//从数据库中取出数据并放到列表list中

List<Student> studentList = new List<Student>();

var query = from s in db.TStudents

select new

{

StudentID = s.StudentID,

Name = s.Name,

Hometown = s.Hometown,

Gender = s.Gender,

Brithday = s.Birthday,

ClassID = s.ClassID,

Weight = s.Weight,

Height = s.Height,

Desc = s.Desc

};

foreach (var item in query)

{

Student student = new Student { StudentID = item.StudentID, Name = item.Name, Hometown = item.Hometown, Gender = item.Gender, Brithday = item.Brithday, ClassID = item.ClassID, Weight = item.Weight, Height = item.Height, Desc = item.Desc };

studentList.Add(student);

}

//基于创建的list使用LINQ to JSON创建期望格式的JSON数据

lbMsg.InnerText = new JObject(

new JProperty("total",studentList.Count),

new JProperty("rows",

new JArray(

//使用LINQ to JSON可直接在select语句中生成JSON数据对象，无须其它转换过程

from p in studentList

select new JObject(

new JProperty("studentID",p.StudentID),

new JProperty("name",p.Name),

new JProperty("homeTown",p.Hometown)

)

)

)

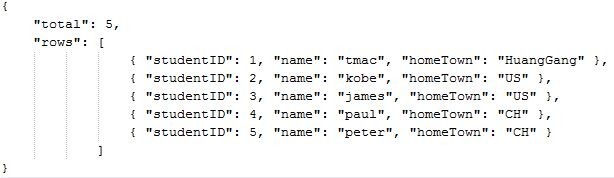
).ToString();

}

}

[复制代码](javascript:void(0);)

输出结果为：

[](http://images.cnitblog.com/blog/311549/201306/10165716-effccc4be3054f5ab8ad12692ad7e613.jpg)

**3，处理客户端提交的JSON数据**

客户端提交过来的数据一般都是json字符串，有了更好地进行操作（面向对象的方式），所以我们一般都会想办法将json字符串转换为json对象。例如客户端提交了以下数组格式json字符串。

[

{StudentID:"100",Name:"aaa",Hometown:"china"},

{StudentID:"101",Name:"bbb",Hometown:"us"},

{StudentID:"102",Name:"ccc",Hometown:"england"}

]

在服务端就可以使用JObject或JArray的Parse方法轻松地将json字符串转换为json对象，然后通过对象的方式提取数据。下面是服务端代码。

[复制代码](javascript:void(0);)

protected void Page\_Load(object sender, EventArgs e)

{

string inputJsonString = @"

[

{StudentID:'100',Name:'aaa',Hometown:'china'},

{StudentID:'101',Name:'bbb',Hometown:'us'},

{StudentID:'102',Name:'ccc',Hometown:'england'}

]";

JArray jsonObj = JArray.Parse(inputJsonString);

string message = @"<table border='1'>

<tr><td width='80'>StudentID</td><td width='100'>Name</td><td width='100'>Hometown</td></tr>";

string tpl = "<tr><td>{0}</td><td>{1}</td><td>{2}</td></tr>";

foreach (JObject jObject in jsonObj)

{

message += String.Format(tpl, jObject["StudentID"], jObject["Name"],jObject["Hometown"]);

}

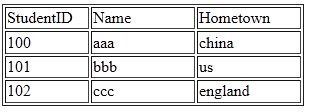
message += "</table>";

lbMsg.InnerHtml = message;

}

[复制代码](javascript:void(0);)

输出结果：

[](http://images.cnitblog.com/blog/311549/201306/10165723-b51ce912fc4a436195a738d0dad0d324.jpg)

**当然，服务端除了使用LINQ to JSON来转换json字符串外，也可以使用JsonConvert的DeserializeObject方法。**如下面代码实现上面同样的功能。

List<Student> studentList = JsonConvert.DeserializeObject<List<Student>>(inputJsonString);//注意这里必须为List<Student>类型,因为客户端提交的是一个数组json

foreach (Student student in studentList)

{

message += String.Format(tpl, student.StudentID, student.Name,student.Hometown);

}

**总结**

在客户端，读写json对象可以使用”.”操作符或”["key”]”，json字符串转换为json对象使用eval()函数。  
在服务端，由.net对象转换json字符串优先使用JsonConvert对象的SerializeObject方法，定制输出json字符串使用LINQ to JSON。由json字符串转换为.net对象优先使用JsonConvert对象的DeserializeObject方法，然后也可以使用LINQ to JSON。