什么是 JSON？

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式，易于阅读和编写，同时也易于机器解析和生成。它基于 ECMA262 语言规范（1999-12 第三版）中 JavaScript 编程语言的一个子集。 JSON 采用与编程语言无关的文本格式，但是也使用了类 C 语言（包括 C， C++， C#， Java， JavaScript， Perl， Python 等）的习惯，这些特性使 JSON 成为理想的数据交换格式。

JSON 与 XML 的比较

◆**可读性**

JSON 和 XML 的可读性可谓不相上下，一边是建议的语法，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，JSON 不能的。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 json.org 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。

JSON 定义

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式，易于阅读和编写，同时也易于机器解析和生成。它基于 ECMA262 语言规范（1999-12 第三版）中 JavaScript 编程语言的一个子集。 JSON 采用与编程语言无关的文本格式，但是也使用了类 C 语言（包括 C， C++， C#， Java， JavaScript， Perl， Python 等）的习惯，这些特性使 JSON 成为理想的数据交换格式。

JSON 的结构基于下面两点

1. "名称/值"对的集合 不同语言中，它被理解为对象(object)，记录(record)，结构(struct)，字典(dictionary)，哈希表(hash table)，键列表(keyed list)等
2. 值的有序列表 多数语言中被理解为数组(array)

JSON 使用：

JSON 以一种特定的字符串形式来表示 JavaScript 对象。如果将具有这样一种形式的字符串赋给任意一个 JavaScript 变量，那么该变量会变成一个对象引用，而这个对象就是字符串所构建出来的，好像有点拗口，我们还是用实例来说明。

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*
这里假设我们需要创建一个 User 对象，并具有以下属性

用户 ID
用户名
用户 Email
您可以使用以下 JSON 形式来表示 User 对象：

{"UserID":11, "Name":"Truly", "Email":"zhuleipro◎hotmail.com"};

然后如果把这一字符串赋予一个 JavaScript 变量，那么就可以直接使用对象的任一属性了。

完整代码:

```
<script>
var User = {"UserID":11, "Name":"Truly", "Email":"zhuleipro◎hotmail.com"};
alert(User.Name);

</script>
```

实际使用时可能更复杂一点，比如我们为 Name 定义更详细的结构，使它具有 FirstName 和 LastName：

```
<script>

{"UserID":11,  "Name":{"FirstName":"Truly","LastName":"Zhu"},  "Email":"zhuleipro ◎ hotmail.com"}

完整代码:

<script>
var User = {"UserID":11, "Name":{"FirstName":"Truly","LastName":"Zhu"}, "Email":"zhuleipro ◎hotmail.com"};
alert(User.Name.FirstName);
```

```
</script>
```

现在我们增加一个新的需求，我们某个页面需要一个用户列表，而不仅仅是一个单一的用户信息，那么这里就需要创建一个用户列表数组。

下面代码演示了使用 JSON 形式定义这个用户列表：

```
[
{"UserID":11,  "Name":{"FirstName":"Truly","LastName":"Zhu"},  "Email":"zhuleipro ◎hotmail.com"},
{"UserID":12,  "Name":{"FirstName":"Jeffrey","LastName":"Richter"},  "Email":"xxx ◎xxx.com"},
{"UserID":13, "Name":{"FirstName":"Scott","LastName":"Gu"}, "Email":"xxx2◎xxx2.com"}
]
```

完整代码:

```
<script>
var UserList = [
{"UserID":11,  "Name":{"FirstName":"Truly","LastName":"Zhu"},  "Email":"zhuleipro ◎hotmail.com"},
{"UserID":12,  "Name":{"FirstName":"Jeffrey","LastName":"Richter"},  "Email":"xxx ◎xxx.com"},
{"UserID":13, "Name":{"FirstName":"Scott","LastName":"Gu"}, "Email":"xxx2◎xxx2.com"}
];
alert(UserList[0].Name.FirstName);

</script>
```

事实上除了使用"."引用属性外，我们还可以使用下面语句：

```
alert(UserList[0]["Name"]["FirstName"]); 或者 alert(UserList[0].Name["FirstName"]);
```

json 的修改：

```
UserList．UserID[0] = “01”
```

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。

完整事例：

```
function jsontest()
{
   //json
   var jsontext = '{"person":{"weight":"75kg","age":"24"},"num":["1","2","3","4"]}';
   //解析 json
   var obj= eval("("+jsontext+")");//通过 eval()  函数可以将 JSON 字符串转化为对象。
    //var jsontext = jsontext.parseJSON();  //第二种转换
   //取出 json 中的信息
   alert(obj.person.weight);
}
```

现在读者应该对 JSON 的使用有点认识了，归纳为以下几点：

1.对象是属性、值对的集合。一个对象的开始于“{”，结束于“}”。每一个属性名和值间用“:”提示，属性间用“,”分隔。
2.数组是有顺序的值的集合。一个数组开始于"["，结束于"]"，值之间用","分隔。
3.值可以是引号里的字符串、数字、true、false、null，也可以是对象或数组。这些结构都能嵌套。
4.字符串和数字的定义和 C#或 Java 基本一致。

Json 应用

```
<script type="text/javascript">
var Person = {
Name:'TerryLee',
Age:25
};

function myObject()
{
return Person;
}
</script>
```

```
public class Person
{
public String Name { get; set; }
public int Age { get; set; }
}
```

```
void btnInvoke_Click(object sender, RoutedEventArgs e)
{
ScriptObject script =
HtmlPage.Window.Invoke("myObject", null) as ScriptObject;

    Person person = script.ConvertTo<Person>();

    this.tblStatus.Text =
String.Format(@"这里是 JavaScript 中的 JSON 对象，Name={0}，Age={1}",
person.Name,
person.Age.ToString());
}
```