## JSON：JavaScript 对象表示法（JavaScript Object Notation）

JSON 是存储和交换文本信息的语法。类似 XML。

JSON 比 XML 更小、更快，更易解析。

每一章中用到的实例

{

"employees": [

{ "firstName":"Bill" , "lastName":"Gates" },

{ "firstName":"George" , "lastName":"Bush" },

{ "firstName":"Thomas" , "lastName":"Carter" }

]

}

这个 employee 对象是包含 3 个员工记录（对象）的数组。

## 什么是 JSON ？

1、JSON 指的是 JavaScript 对象表示法（**J**ava**S**cript **O**bject **N**otation）

2、JSON 是轻量级的文本数据交换格式

3、JSON 独立于语言 \*

4、JSON 具有自我描述性，更易理解

\* JSON 使用 JavaScript 语法来描述数据对象，但是 JSON 仍然独立于语言和平台。JSON 解析器和 JSON 库支持许多不同的编程语言。

## JSON - 转换为 JavaScript 对象

JSON 文本格式在语法上与创建 JavaScript 对象的代码相同。

由于这种相似性，无需解析器，JavaScript 程序能够使用内建的 [eval() 函数](http://www.w3school.com.cn/jsref/jsref_eval.asp)，用 JSON 数据来生成原生的 JavaScript 对象。

## 实例

通过我们的编辑器，您可以在线编辑 JavaScript 代码，然后通过点击一个按钮来查看结果：

<html>

<body>

<h2>在 JavaScript 中创建 JSON 对象</h2>

<p>

Name: <span id="jname"></span><br />

Age: <span id="jage"></span><br />

Address: <span id="jstreet"></span><br />

Phone: <span id="jphone"></span><br />

</p>

<script type="text/javascript">

var JSONObject= {

"name":"Bill Gates",

"street":"Fifth Avenue New York 666",

"age":56,

"phone":"555 1234567"};

document.getElementById("jname").innerHTML=JSONObject.name

document.getElementById("jage").innerHTML=JSONObject.age

document.getElementById("jstreet").innerHTML=JSONObject.street

document.getElementById("jphone").innerHTML=JSONObject.phone

</script>

</body>

</html>

## Json组成

Var json = {“aaa”: 1,“bbb”: 2,“ccc”: 3,“ddd”: 4}

Json由{}和key:value以及逗号组成，三部分。（只有一个键值对key:value时,可以没有逗号）

## For...in...

Var json = {“aaa”: 1,“bbb”: 2,“ccc”: 3,“ddd”: 4}

for(var key in json){

//key代表aaa,bbb.....等

//json[key]代表1,2,3....等

}

## 参数和传值问题

1. 简单类型数据做参数，函数内部对参数的修改不应影响外部变量

简单类型传数值。

1. 复杂类型数据做参数，函数内部对参数的修改会应影响外部变量

复杂类型传地址。

## 类似 XML

JSON 是纯文本

JSON 具有“自我描述性”（人类可读）

JSON 具有层级结构（值中存在值）

JSON 可通过 JavaScript 进行解析

JSON 数据可使用 AJAX 进行传输

## 相比 XML 的不同之处

没有结束标签

更短

读写的速度更快

能够使用内建的 JavaScript eval() 方法进行解析

使用数组

不使用保留字

## 为什么使用 JSON？

对于 AJAX 应用程序来说，JSON 比 XML 更快更易使用：

### 使用 XML

读取 XML 文档

使用 XML DOM 来循环遍历文档

读取值并存储在变量中

### 使用 JSON

读取 JSON 字符串

用 eval() 处理 JSON 字符串

# JSON 语法规则

JSON 语法是 JavaScript 对象表示法语法的子集。

## JSON 名称/值对

JSON 数据的书写格式是：名称/值对。

名称/值对包括字段名称（在双引号中），后面写一个冒号，然后是值：

"firstName" : "John"

这很容易理解，等价于这条 JavaScript 语句：

firstName = "John"

## JSON 值

JSON 值可以是：

1、数字（整数或浮点数）

2、字符串（在双引号中）

3、逻辑值（true 或 false）

4、数组（在方括号中）

5、对象（在花括号中）

6、null

## JSON 对象

JSON 对象在花括号中书写：

对象可以包含多个名称/值对：

{ "firstName":"John" , "lastName":"Doe" }

这一点也容易理解，与这条 JavaScript 语句等价：

firstName = "John"

lastName = "Doe"

## JSON 数组

JSON 数组在方括号中书写：

数组可包含多个对象：

{

"employees": [

{ "firstName":"John" , "lastName":"Doe" },

{ "firstName":"Anna" , "lastName":"Smith" },

{ "firstName":"Peter" , "lastName":"Jones" }

]

}

在上面的例子中，对象 "employees" 是包含三个对象的数组。每个对象代表一条关于某人（有姓和名）的记录。

## JSON 使用 JavaScript 语法

因为 JSON 使用 JavaScript 语法，所以无需额外的软件就能处理 JavaScript 中的 JSON。

通过 JavaScript，您可以创建一个对象数组，并像这样进行赋值：

var employees = [

{ "firstName":"Bill" , "lastName":"Gates" },

{ "firstName":"George" , "lastName":"Bush" },

{ "firstName":"Thomas" , "lastName": "Carter" }

];

可以像这样访问 JavaScript 对象数组中的第一项：

employees[0].lastName;

返回的内容是：

Gates

可以像这样修改数据：

employees[0].lastName = "Jobs";

## JSON 文件

1、JSON 文件的文件类型是 ".json"

2、JSON 文本的 MIME 类型是 "application/json"

# JSON 使用

## 把 JSON 文本转换为 JavaScript 对象

JSON 最常见的用法之一，是从 web 服务器上读取 JSON 数据（作为文件或作为 HttpRequest），将 JSON 数据转换为 JavaScript 对象，然后在网页中使用该数据。

为了更简单地为您讲解，我们使用字符串作为输入进行演示（而不是文件）。

## JSON 实例 - 来自字符串的对象

创建包含 JSON 语法的 JavaScript 字符串：

var txt = '{ "employees" : [' +

'{ "firstName":"Bill" , "lastName":"Gates" },' +

'{ "firstName":"George" , "lastName":"Bush" },' +

'{ "firstName":"Thomas" , "lastName":"Carter" } ]}';

由于 JSON 语法是 JavaScript 语法的子集，JavaScript 函数 eval() 可用于将 JSON 文本转换为 JavaScript 对象。

eval() 函数使用的是 JavaScript 编译器，可解析 JSON 文本，然后生成 JavaScript 对象。必须把文本包围在括号中，这样才能避免语法错误：

var obj = eval ("(" + txt + ")");

在网页中使用 JavaScript 对象：

<p>

First Name: <span id="fname"></span><br />

Last Name: <span id="lname"></span><br />

</p>

<script type="text/javascript">

document.getElementById("fname").innerHTML = obj.employees[1].firstName

document.getElementById("lname").innerHTML = obj.employees[1].lastName

</script>

## JSON 解析器

**提示：**eval() 函数可编译并执行任何 JavaScript 代码。这隐藏了一个潜在的安全问题。

使用 JSON 解析器将 JSON 转换为 JavaScript 对象是更安全的做法。JSON 解析器只能识别 JSON 文本，而不会编译脚本。

在浏览器中，这提供了原生的 JSON 支持，而且 JSON 解析器的速度更快。

较新的浏览器和最新的 ECMAScript (JavaScript) 标准中均包含了原生的对 JSON 的支持。

|  |  |
| --- | --- |
| Web 浏览器支持 | Web 软件支持 |
| Firefox (Mozilla) 3.5  Internet Explorer 8  Chrome  Opera 10  Safari 4 | jQuery  Yahoo UI  Prototype  Dojo  ECMAScript 1.5 |

<html>

<body>

<h2>通过 JSON 字符串来创建对象</h3>

<p>

First Name: <span id="fname"></span><br />

Last Name: <span id="lname"></span><br />

</p>

<script type="text/javascript">

var txt = '{"employees":[' +

'{"firstName":"Bill","lastName":"Gates" },' +

'{"firstName":"George","lastName":"Bush" },' +

'{"firstName":"Thomas","lastName":"Carter" }]}';

obj = JSON.parse(txt);

document.getElementById("fname").innerHTML=obj.employees[1].firstName

document.getElementById("lname").innerHTML=obj.employees[1].lastName

</script>

</body>

</html>