AJAX 教程

AJAX = Asynchronous JavaScript and XML（异步的 JavaScript 和 XML）。

AJAX 不是新的编程语言，而是一种使用现有标准的新方法。

AJAX 最大的优点是在不重新加载整个页面的情况下，可以与服务器交换数据并更新部分网页内容。

AJAX 不需要任何浏览器插件，但需要用户允许JavaScript在浏览器上执行。

AJAX 实例

使用 AJAX 修改该文本内容

修改内容

尝试一下 »

现在开始学习 AJAX！

阅读本教程前，您需要了解的知识：

阅读本教程，您需要有以下基础：

- HTML 和 CSS 基础
- JavaScript 基础

如果您想学习这些基础知识，您可以在我们的首页找到相应的教程菜鸟教程。

AJAX 应用

- 运用XHTML+CSS来表达资讯；
- 运用JavaScript操作DOM（Document Object Model）来执行动态效果；
- 运用XML和XSLT操作资料；
- 运用XMLHttpRequest或新的Fetch API与网页服务器进行异步资料交换；
- 注意：AJAX与Flash、Silverlight和Java Applet等RIA技术是有区分的。

AJAX 简介

AJAX是一种在无需重新加载整个网页的情况下，能够更新部分网页的技术。

您应当具备的基础知识

在继续学习之前，您需要对下面的知识有基本的了解：

- HTML / XHTML
- CSS
- JavaScript / DOM

如果您希望首先学习这些项目，请在我们的首页访问这些教程。

什么是 AJAX？

AJAX = 异步 JavaScript 和 XML。

AJAX是一种用于创建快速动态网页的技术。

通过在后台与服务器进行少量数据交换，AJAX可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

传统的网页（不使用 AJAX）如果需要更新内容，必需重载整个网页面。

有很多使用 AJAX的应用程序案例：新浪微博、Google 地图、开心网等等。

AJAX 工作原理

AJAX

AJAX是基于现有的Internet标准

AJAX是基于现有的Internet标准，并且联合使用它们：

- XMLHttpRequest 对象 (异步的与服务器交换数据)
- JavaScript/DOM (信息显示/交互)
- CSS (给数据定义样式)
- XML (作为转换数据的格式)

AJAX应用程序与浏览器和平台无关的！

Google Suggest

在 2005 年，Google 通过其 Google Suggest 使 AJAX变得流行起来。

Google Suggest 使用 AJAX创造出动态性极强的 web 界面：当您在谷歌的搜索框输入关键字时，JavaScript 会把这些字符发送到服务器，然后服务器会返回一个搜索建议的列表。

今天就开始使用 AJAX

AJAX基于已有的标准。这些标准已被大多数开发者使用多年。

请阅读下一章，看看 AJAX是如何工作的！

AJAX 实例

为了帮助您理解 AJAX 的工作原理，我们创建了一个小型的 AJAX 应用程序:

实例

使用 **AJAX** 修改该文本内容

修改内容

尝试一下 »

AJAX 实例解析

上面的 AJAX 应用程序包含一个 div 和一个按钮。

div 部分用于显示来自服务器的信息。当按钮被点击时，它负责调用名为 loadXMLDoc() 的函数：

```
<div id="myDiv"><h2>使用 AJAX 修改该文本内容</h2></div>
<button type="button" onclick="loadXMLDoc()">修改内容</button>
```

接下来，在页面的 head 部分添加一个 <script> 标签。该标签中包含了这个 loadXMLDoc() 函数：

```
<head>
<script>
function loadXMLDoc()
{
.... AJAX 脚本执行 ...
}
</script>
</head>
```

下面的章节会为您讲解 AJAX 的工作原理。

反馈/建议

Copyright © 2013-2018 菜鸟教程 runoob.com All Rights Reserved. 备案号：闽ICP备15012807号-1

AJAX - 创建 XMLHttpRequest 对象

XMLHttpRequest 是 AJAX 的基础。

XMLHttpRequest 对象

所有现代浏览器均支持 XMLHttpRequest 对象（IE5 和 IE6 使用 ActiveXObject）。

XMLHttpRequest 用于在后台与服务器交换数据。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

创建 XMLHttpRequest 对象

所有现代浏览器（IE7+、Firefox、Chrome、Safari 以及 Opera）均内建 XMLHttpRequest 对象。

创建 XMLHttpRequest 对象的语法：

```
variable=new XMLHttpRequest();
```

老版本的 Internet Explorer （IE5 和 IE6）使用 ActiveX 对象：

```
variable=new ActiveXObject("Microsoft.XMLHTTP");
```

为了应对所有的现代浏览器，包括 IE5 和 IE6，请检查浏览器是否支持 XMLHttpRequest 对象。如果支持，则创建 XMLHttpRequest 对象。如果不支持，则创建 ActiveXObject ：

实例

```
var xmlhttp;
if (window.XMLHttpRequest)
{
// IE7+, Firefox, Chrome, Opera, Safari 浏览器执行代码
xmlhttp=new XMLHttpRequest();
}
else
{
// IE6, IE5 浏览器执行代码
xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
}
```

尝试一下 »

在下一章中，您将学习发送服务器请求的知识。

AJAX XMLHttpRequest 服务器响应 AJAX 创建 XMLHttpRequest 对象

AJAX - 向服务器发送请求请求

XMLHttpRequest 对象用于和服务器交换数据。

向服务器发送请求

如需将请求发送到服务器，我们使用 XMLHttpRequest 对象的 open() 和 send() 方法：

```
xmlhttp.open("GET","ajax_info.txt",true);
xmlhttp.send();
```

方法	描述
open(*method*,*url*,*async*)	规定请求的类型、URL 以及是否异步处理请求。 *method*：请求的类型；GET 或 POST *url*：文件在服务器上的位置 *async*：true（异步）或 false（同步）
send(*string*)	将请求发送到服务器。 *string*：仅用于 POST 请求

GET 还是 POST?

与 POST 相比，GET 更简单也更快，并且在大部分情况下都能用。

然而，在以下情况中，请使用 POST 请求：

无法使用缓存文件（更新服务器上的文件或数据库）

向服务器发送大量数据（POST 没有数据量限制）

发送包含未知字符的用户输入时，POST 比 GET 更稳定也更可靠

GET 请求

一个简单的 GET 请求：

实例

```
xmlhttp.open("GET","/try/ajax/demo_get.php",true);
xmlhttp.send();
```

尝试一下 »

在上面的例子中，您可能得到的是缓存的结果。

为了避免这种情况，请向 URL 添加一个唯一的 ID：

实例

```
xmlhttp.open("GET","/try/ajax/demo_get.php?t=" + Math.random(),true);
xmlhttp.send();
```

尝试一下 »

如果您希望通过 GET 方法发送信息，请向 URL 添加信息：

实例

```
xmlhttp.open("GET","/try/ajax/demo_get2.php?fname=Henry&lname=Ford",true);
xmlhttp.send();
```

尝试一下 »

POST 请求

一个简单 POST 请求：

实例

```
xmlhttp.open("POST","/try/ajax/demo_post.php",true);
xmlhttp.send();
```

尝试一下 »

如果需要像 HTML 表单那样 POST 数据，请使用 setRequestHeader() 来添加 HTTP 头。然后在 send() 方法中规定您希望发送的数据：

实例

```
xmlhttp.open("POST","/try/ajax/demo_post2.php",true);
xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");
xmlhttp.send("fname=Henry&lname=Ford");
```

尝试一下 »

方法	描述
setRequestHeader(*header*,*value*)	向请求添加 HTTP 头。 *header*: 规定头的名称 *value*: 规定头的值

url - 服务器上的文件

open() 方法的 *url* 参数是服务器上文件的地址：

```
xmlhttp.open("GET","ajax_test.html",true);
```

该文件可以是任何类型的文件，比如 .txt 和 .xml，或者服务器脚本文件，比如 .asp 和 .php （在传回响应之前，能够在服务器上执行任务）。

异步 - True 或 False?

AJAX 指的是异步 JavaScript 和 XML（Asynchronous JavaScript and XML）。

XMLHttpRequest 对象如果要用于 AJAX 的话，其 open() 方法的 async 参数必须设置为 true：

```
xmlhttp.open("GET","ajax_test.html",true);
```

对于 web 开发人员来说，发送异步请求是一个巨大的进步。很多在服务器执行的任务都相当费时。AJAX 出现之前，这可能会引起应用程序挂起或停止。

通过 AJAX，JavaScript 无需等待服务器的响应，而是：

在等待服务器响应时执行其他脚本

当响应就绪后对响应进行处理

Async=true

当使用 async=true 时，请规定在响应处于 onreadystatechange 事件中的就绪状态时执行的函数：

实例

```
xmlhttp.onreadystatechange=function()
{
if (xmlhttp.readyState==4 && xmlhttp.status==200)
```

```
{
document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
}
}
xmlhttp.open("GET","/try/ajax/ajax_info.txt",true);
xmlhttp.send();
```

尝试一下 »

您将在稍后的章节学习更多有关 onreadystatechange 的内容。

Async = false

如需使用 async=false，请将 open() 方法中的第三个参数改为 false：

```
xmlhttp.open("GET","test1.txt",false);
```

我们不推荐使用 async=false，但是对于一些小型的请求，也是可以的。

请记住，JavaScript 会等到服务器响应就绪才继续执行。如果服务器繁忙或缓慢，应用程序会挂起或停止。

注意： 当您使用 async=false 时，请不要编写 onreadystatechange 函数 - 把代码放到 send() 语句后面即可：

实例

```
xmlhttp.open("GET","/try/ajax/ajax_info.txt",false);
xmlhttp.send();
document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
```

尝试一下 »

AJAX XMLHttpRequest 服务器响应　　AJAX 创建 XMLHttpRequest 对象

点我分享笔记

反馈/建议

AJAX – onreadystatechange 事件　　AJAX – 向服务器发送请求

AJAX - 服务器 响应

服务器响应

如需获得来自服务器的响应，请使用 XMLHttpRequest 对象的 responseText 或 responseXML 属性。

属性	描述
responseText	获得字符串形式的响应数据。
responseXML	获得 XML 形式的响应数据。

responseText 属性

如果来自服务器的响应并非 XML，请使用 responseText 属性。

responseText 属性返回字符串形式的响应，因此您可以这样使用：

实例

```
document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
```

尝试一下 »

responseXML 属性

如果来自服务器的响应是 XML，而且需要作为 XML 对象进行解析，请使用 responseXML 属性：

实例

请求 cd_catalog.xml 文件，并解析响应：

```
xmlDoc=xmlhttp.responseXML;
txt="";
x=xmlDoc.getElementsByTagName("ARTIST");
for (i=0;i<x.length;i++)
{
txt=txt + x[i].childNodes[0].nodeValue + "<br>";
}
document.getElementById("myDiv").innerHTML=txt;
```

尝试一下 »

AJAX – onreadystatechange 事件　　AJAX – 向服务器发送请求

点我分享笔记

反馈/建议

AJAX ASP/PHP　　AJAX XMLHttpRequest 服务器响应

AJAX - onreadystatechange 事件

onreadystatechange 事件

当请求被发送到服务器时，我们需要执行一些基于响应的任务。

每当 readyState 改变时，就会触发 onreadystatechange 事件。

readyState 属性存有 XMLHttpRequest 的状态信息。

下面是 XMLHttpRequest 对象的三个重要的属性：

属性	描述

onreadystatechange	存储函数（或函数名），每当 readyState 属性改变时，就会调用该函数。
readyState	存有 XMLHttpRequest 的状态。从 0 到 4 发生变化。 0: 请求未初始化 1: 服务器连接已建立 2: 请求已接收 3: 请求处理中 4: 请求已完成，且响应已就绪
status	200: "OK" 404: 未找到页面

在 onreadystatechange 事件中，我们规定当服务器响应已做好被处理的准备时所执行的任务。

当 readyState 等于 4 且状态为 200 时，表示响应已就绪：

实例

```
xmlhttp.onreadystatechange=function()
{
if (xmlhttp.readyState==4 && xmlhttp.status==200)
{
document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
}
}
```

尝试一下 »

注意： onreadystatechange 事件被触发 4 次（0 - 4），分别是： 0-1、1-2、2-3、3-4，对应着 readyState 的每个变化。

使用回调函数

回调函数是一种以参数形式传递给另一个函数的函数。

如果您的网站上存在多个 AJAX 任务，那么您应该为创建 XMLHttpRequest 对象编写一个*标准*的函数，并为每个 AJAX 任务调用该函数。

该函数调用应该包含 URL 以及发生 onreadystatechange 事件时执行的任务（每次调用可能不尽相同）：

实例

```
function myFunction()
{
loadXMLDoc("/try/ajax/ajax_info.txt",function()
{
if (xmlhttp.readyState==4 && xmlhttp.status==200)
{
document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
}
});
}
```

尝试一下 »

□ AJAX ASP/PHP　　AJAX XMLHttpRequest 服务器响应 □

AJAX ASP/PHP 实例

AJAX用于创造动态性更强的应用程序。

AJAX ASP/PHP 实例

下面的例子将为您演示当用户在输入框中键入字符时，网页如何与 web 服务器进行通信： 请在下面的输入框中键入字母（A - Z）：

实例

在输入框中尝试输入字母 a:

输入姓名:

提示信息:

尝试一下 »

实例解析 - showHint() 函数

当用户在上面的输入框中键入字符时，会执行函数 "showHint()" 。该函数由 "onkeyup" 事件触发：

```
function showHint(str)
{
var xmlhttp;
if (str.length==0)
{
document.getElementById("txtHint").innerHTML="";
return;
}
if (window.XMLHttpRequest)
{
// IE7+, Firefox, Chrome, Opera, Safari 浏览器执行代码
xmlhttp=new XMLHttpRequest();
}
else
{
// IE6, IE5 浏览器执行代码
xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
}
xmlhttp.onreadystatechange=function()
{
if (xmlhttp.readyState==4 && xmlhttp.status==200)
{
document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
}
}
xmlhttp.open("GET","/try/ajax/gethint.php?q="+str,true);
xmlhttp.send();
}
```

源代码解析：

如果输入框为空 (str.length==0)，则该函数清空 txtHint 占位符的内容，并退出函数。

如果输入框不为空，showHint() 函数执行以下任务：

创建 XMLHttpRequest 对象

当服务器响应就绪时执行函数

把请求发送到服务器上的文件

请注意我们向 URL 添加了一个参数 q（带有输入框的内容）

AJAX 服务器页面 - ASP 和 PHP

由上面的 JavaScript 调用的服务器页面是 ASP 文件，名为 "gethint.asp"。

下面，我们创建了两个版本的服务器文件，一个用 ASP 编写，另一个用 PHP 编写。

ASP 文件

"gethint.asp" 中的源代码会检查一个名字数组，然后向浏览器返回相应的名字：

```
<%

response.expires=-1

dim a(30)

'Fill up array with names

a(1)="Anna"

a(2)="Brittany"

a(3)="Cinderella"

a(4)="Diana"

a(5)="Eva"

a(6)="Fiona"

a(7)="Gunda"

a(8)="Hege"

a(9)="Inga"

a(10)="Johanna"

a(11)="Kitty"

a(12)="Linda"

a(13)="Nina"

a(14)="Ophelia"

a(15)="Petunia"

a(16)="Amanda"

a(17)="Raquel"

a(18)="Cindy"

a(19)="Doris"

a(20)="Eve"
```

```
a(21)="Evita"

a(22)="Sunniva"

a(23)="Tove"

a(24)="Unni"

a(25)="Violet"

a(26)="Liza"

a(27)="Elizabeth"

a(28)="Ellen"

a(29)="Wenche"

a(30)="Vicky"



'get the q parameter from URL

q=ucase(request.querystring("q"))



'lookup all hints from array if length of q>0

if len(q)>0 then

  hint=""

  for i=1 to 30

    if q=ucase(mid(a(i),1,len(q))) then

      if hint="" then

        hint=a(i)

      else

        hint=hint & " , " & a(i)

      end if

    end if

  next

end if



'Output "no suggestion" if no hint were found

'or output the correct values

if hint="" then

  response.write("no suggestion")

else
```

```
  response.write(hint)

end if

%>
```

PHP 文件

下面的代码用 PHP 编写，与上面的 ASP 代码作用是一样的。

```
<?php

// Fill up array with names

$a[]="Anna";

$a[]="Brittany";

$a[]="Cinderella";

$a[]="Diana";

$a[]="Eva";

$a[]="Fiona";

$a[]="Gunda";

$a[]="Hege";

$a[]="Inga";

$a[]="Johanna";

$a[]="Kitty";

$a[]="Linda";

$a[]="Nina";

$a[]="Ophelia";

$a[]="Petunia";

$a[]="Amanda";

$a[]="Raquel";

$a[]="Cindy";

$a[]="Doris";

$a[]="Eve";

$a[]="Evita";

$a[]="Sunniva";

$a[]="Tove";
```

```
$a[]="Unni";

$a[]="Violet";

$a[]="Liza";

$a[]="Elizabeth";

$a[]="Ellen";

$a[]="Wenche";

$a[]="Vicky";



//get the q parameter from URL

$q=$_GET["q"];



//lookup all hints from array if length of q>0

if (strlen($q) > 0)

{

  $hint="";

  for($i=0; $i<count($a); $i++)

  {

    if (strtolower($q)==strtolower(substr($a[$i],0,strlen($q))))

    {

      if ($hint=="")

      {

        $hint=$a[$i];

      }

      else

      {

        $hint=$hint." , ".$a[$i];

      }

    }

  }

}



// Set output to "no suggestion" if no hint were found

// or to the correct values
```

```
if ($hint == "")
{
  $response="no suggestion";
}
else
{
  $response=$hint;
}

//output the response
echo $response;
?>
```

□ AJAX 数据库　　AJAX – onreadystatechange 事件 □

□

1 篇笔记

□ 写笔记

#1

□

xmlhttp.readyState的值及解释：

0：请求未初始化（还没有调用 open()）。
1：请求已经建立，但是还没有发送（还没有调用 send()）。
2：请求已发送，正在处理中（通常现在可以从响应中获取内容头）。
3：请求在处理中；通常响应中已有部分数据可用了，但是服务器还没有完成响应的生成。
4：响应已完成；您可以获取并使用服务器的响应了。

xmlhttp.status的值及解释：

100——客户必须继续发出请求
101——客户要求服务器根据请求转换HTTP协议版本
200——交易成功
201——提示知道新文件的URL
202——接受和处理、但处理未完成
203——返回信息不确定或不完整
204——请求收到，但返回信息为空
205——服务器完成了请求，用户代理必须复位当前已经浏览过的文件
206——服务器已经完成了部分用户的GET请求
300——请求的资源可在多处得到
301——删除请求数据
302——在其他地址发现了请求数据
303——建议客户访问其他URL或访问方式
304——客户端已经执行了GET，但文件未变化
305——请求的资源必须从服务器指定的地址得到
306——前一版本HTTP中使用的代码，现行版本中不再使用
307——申明请求的资源临时性删除
400——错误请求，如语法错误
401——请求授权失败
402——保留有效ChargeTo头响应
403——请求不允许
404——没有发现文件、查询或URl
405——用户在Request-Line字段定义的方法不允许
406——根据用户发送的Accept拖，请求资源不可访问
407——类似401，用户必须首先在代理服务器上得到授权
408——客户端没有在用户指定的饿时间内完成请求
409——对当前资源状态，请求不能完成

410——服务器上不再有此资源且无进一步的参考地址
411——服务器拒绝用户定义的Content-Length属性请求
412——一个或多个请求头字段在当前请求中错误
413——请求的资源大于服务器允许的大小
414——请求的资源URL长于服务器允许的长度
415——请求资源不支持请求项目格式
416——请求中包含Range请求头字段，在当前请求资源范围内没有range指示值，请求也不包含If-Range请求头字段
417——服务器不满足请求Expect头字段指定的期望值，如果是代理服务器，可能是下一级服务器不能满足请求
合起来
500——服务器产生内部错误
501——服务器不支持请求的函数
502——服务器暂时不可用，有时是为了防止发生系统过载
503——服务器过载或暂停维修
504——关口过载，服务器使用另一个关口或服务来响应用户，等待时间设定值较长
505——服务器不支持或拒绝支请求头中指定的HTTP版本
1xx信息响应类，表示接收到请求并且继续处理
2xx处理成功响应类，表示动作被成功接收、理解和接受
3xx重定向响应类，为了完成指定的动作，必须接受进一步处理
4xx客户端错误，客户请求包含语法错误或者是不能正确执行
5xx服务端错误，服务器不能正确执行一个正确的请求
`xmlhttp.readyState==4 && xmlhttp.status==200`的解释：请求完成并且成功返回

aa 11个月前 **(11-09)**

反馈/建议

AJAX Database 实例

AJAX可用来与数据库进行动态通信。

AJAX 数据库实例

下面的例子将演示网页如何通过 AJAX 从数据库读取信息： 请在下面的下拉列表中选择一个客户：

实例

Apple Computer, Inc.

客户信息将显示在这...

尝试一下 »

实例解释 - showCustomer() 函数

当用户在上面的下拉列表中选择某个客户时，会执行名为 "showCustomer()" 的函数。该函数由 "onchange" 事件触发：

```
function showCustomer(str)
{
var xmlhttp;
if (str=="")
{
document.getElementById("txtHint").innerHTML="";
return;
}
```

```
if (window.XMLHttpRequest)
{
// IE7+, Firefox, Chrome, Opera, Safari 浏览器执行代码
xmlhttp=new XMLHttpRequest();
}
else
{
// IE6, IE5 浏览器执行代码
xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
}
xmlhttp.onreadystatechange=function()
{
if (xmlhttp.readyState==4 && xmlhttp.status==200)
{
document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
}
}
xmlhttp.open("GET","/try/ajax/getcustomer.php?q="+str,true);
xmlhttp.send();
}
```

showCustomer() 函数执行以下任务：

- 检查是否已选择某个客户
- 创建 XMLHttpRequest 对象
- 当服务器响应就绪时执行所创建的函数
- 把请求发送到服务器上的文件
- 请注意我们向 URL 添加了一个参数 q（带有输入域中的内容）

AJAX 服务器页面

由上面的 JavaScript 调用的服务器页面是 PHP 文件，名为 "getcustomer.php"。

用 PHP 编写服务器文件也很容易，或者用其他服务器语言。请看用 PHP 编写的相应的例子。

"getcustomer.php" 中的源代码负责对数据库进行查询，然后用 HTML 表格返回结果：

```
<%

response.expires=-1

sql="SELECT * FROM CUSTOMERS WHERE CUSTOMERID="

sql=sql & "'" & request.querystring("q") & "'"



set conn=Server.CreateObject("ADODB.Connection")

conn.Provider="Microsoft.Jet.OLEDB.4.0"

conn.Open(Server.Mappath("/db/northwind.mdb"))

set rs=Server.CreateObject("ADODB.recordset")

rs.Open sql,conn



response.write("<table>")

do until rs.EOF

  for each x in rs.Fields
```

```
    response.write("<tr><td><b>" & x.name & "</b></td>")
    response.write("<td>" & x.value & "</td></tr>")
  next
  rs.MoveNext
loop
response.write("</table>")
%>
```

AJAX XML

AJAX ASP/PHP

点我分享笔记

反馈/建议

AJAX XML 实例

AJAX可用来与 XML 文件进行交互式通信。

AJAX XML 实例

下面的例子将演示网页如何使用 AJAX来读取来自 XML 文件的信息：

实例

获取我收藏的 CD

尝试一下 »

实例解析 - loadXMLDoc() 函数

当用户点击上面的"获取我收藏的 CD"这个按钮，就会执行 loadXMLDoc() 函数。

loadXMLDoc() 函数创建 XMLHttpRequest 对象，添加当服务器响应就绪时执行的函数，并将请求发送到服务器。

当服务器响应就绪时，会构建一个 HTML 表格，从 XML 文件中提取节点（元素），最后使用 XML 数据的 填充 id="demo" 的表格元素：

异步加载 XML 文档

```
function loadXMLDoc() {
var xhttp = new XMLHttpRequest();
```

```
xhttp.onreadystatechange = function() {
if (this.readyState == 4 && this.status == 200) {
myFunction(this);
}
};
xhttp.open("GET", "cd_catalog.xml", true);
xhttp.send();
}
function myFunction(xml) {
var i;
var xmlDoc = xml.responseXML;
var table="<tr><th>Artist</th><th>Title</th></tr>";
var x = xmlDoc.getElementsByTagName("CD");
for (i = 0; i <x.length; i++) {
table += "<tr><td>" +
x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
"</td><td>" +
x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +
"</td></tr>";
}
document.getElementById("demo").innerHTML = table;
}
```

AJAX 服务器页面

上面这个例子中使用的服务器页面实际上是一个名为 "cd_catalog.xml" XML 文件。

AJAX 教程 AJAX 数据库

点我分享笔记

反馈/建议

AJAX 实例

AJAX 实例

实例

AJAX 实例

一个简单的AJAX实例

创建一个简单的XMLHttpRequest，从一个TXT文件中返回数据。

用AJAX加载 XML 文件

创建一个简单的XMLHttpRequest，从一个XML文件中返回数据。

用AJAX进行一次 HEAD 请求

检索资源（文件）的头信息。

用AJAX进行一次指定的 HEAD 请求

检索资源（文件）的指定头信息。

用AJAX从ASP 文件返回数据

当用户在文本框内键入字符时网页如何与Web服务器进行通信

用AJAX从数据库返回数据

用AJAX网页如何获取数据库中的信息

用AJAX从XML 文件返回数据

创建一个XMLHttpRequest从XML文件中检索数据并显示在一个HTML表格中。

用callback函数的AJAX实例

用一个callback函数创建一个XMLHttpRequest，并从一个TXT文件中检索数据。

AJAX 实例

点我分享笔记

反馈/建议