بسم الله الرحمن الرحیم

آموزش AJAX

ارائه شده به : دکتر حسن حقیقی

توسط : سیده زهرا حسینی و سیده الهه جلمبادانی

تیر ۹۰

فهرست

Ajax 4
هدف AJAX 5
شئ XMLHttpRequest 6
اولين برنامه AJAX 7
سنگ بناي AJAX 7
خصوصيت readyState 10
ارسال درخواست به Server 11
كد ASP سمت Server 12
كد AJAX مربوط به فايل HTML 17
كد AJAX مربوط به فايل JavaScript 18
AJAX و بانك‌هاي اطلاعاتي 27
AJAX JavaScript 28
AJAX و تعامل با XML 31

Ajax

AJAX مخفف Asynchronous JavaScript And XML است.

AJAX نوعی برنامه‌نویسی است که در سال ۲۰۰۵ توسط Google و به پیشنهاد آن، فراگیر شد.

AJAX یک زبان برنامه‌نویسی جدید نیست، بلکه یک روش جدید برای استفاده از استانداردهای موجود است.

با AJAX می‌توانید برنامه‌های کاربردی تحت Web بهتر، سریعتر و کاربرپسندانه‌تر بسازید.

AJAX برپایه JavaScript و درخواست‌های HTTP بنا شده است.

قبل از اینکه بتوانید به راحتی باAJAX کار کنید، لازم است که آشنایی متوسطی باHTML/XHTM و همچنین JavaScript داشته باشید.

AJAX یک تکنیک برای ساختن صفحات Web است به‌نحوی که این صفحات، بیشتر با کاربر به‌صورت فعل و انفعالی کار کنند و مشابه برنامه‌های تحت ویندوز عمل‌نمایند.

توسط AJAX، کدهای JavaScript شما می‌توانند مستقیماً با Server به‌وسیله شئ XMLHttpRequest ارتباط برقرار کنند. توسط این شئ، می‌توانید داده‌ها را با یک Web Server تبادل‌نمایید، بدون‌آنکه نیاز به بارگذاری مجدد صفحه داشته‌باشید.

AJAX از انتقال داده‌های غیرهم‌زمان (درخواست‌های HTTP) بین مرورگر و Server استفاده می‌کند، که به صفحات Web اجازه می‌دهد به‌جای کل صفحه، تنها درخواست مقادیر کوچک اطلاعات از Server داشته‌باشند.

تکنیک AJAX برنامه‌های Internet را کوچک‌تر، سریع‌تر و کاربرپسندانه‌تر می‌کند.

AJAX یک تکنولوژی مرورگر است که به نرم‌افزار Web Server وابسته نیست.

AJAX برپایه استانداردهای Web زیر بنا شده است :

JavaScript

XML

HTML

CSS

این استانداردهای Web که در AJAX مورداستفاده قرار گرفته‌اند، به‌خوبی در تمامی مرورگرهای مطرح ازجمله Internet Explorer ۵+، Safari ۱.۲، Mozilla ۱.۰/FireFox، Opera ۸+، NetScape ۷ و... تعریف و پشتیبانی شده‌اند. همچنین برنامه‌های AJAX وابسته به مرورگر و سیستم‌عامل نیستند و به‌خوبی برروی سیستم‌های عامل مختلف ازقبیل Windows، Linux، Lindows، UNIX، FreeBSD، MacOS و... کار می‌کنند.

هدف AJAX، برنامه‌های کاربردی بهتر در محیطInternet است.

برنامه‌های کاربردی تحت Web مزایای متعددی نسبت‌به برنامه‌های کاربردی مجزا از شبکه (Desktop) دارند؛ می‌توانند دایره مخاطبین بیشتری را دربر بگیرند، نصب و پشتیبانی آنها ساده‌تر است، و همچنین راحت‌تر توسعه داده می‌شوند.

هرچند، برنامه‌های اینترنت همیشه به‌اندازه برنامه‌های سنتی رومیزی (Desktop)، کاربرپسندانه و غنی نیستند (به‌دلیل محدودیت منابع و سرعت).

به‌وسیله AJAX، برنامه‌های اینترنت را می‌توان غنی‌تر و کاربرپسندانه‌تر نمود.

درواقع هیچ چیز جدیدی برای یادگیری وجود ندارد. AJAX برمبنای استانداردهای موجود طراحی شده‌است. این استانداردها توسط اغلب طراحان، سال‌های زیادی مورداستفاده قرارگرفته‌اند.

AJAX از **درخواست‌های HTTP استفاده می‌کند :**

در کدنویسی سنتی JavaScript، اگر می‌خواهید اطلاعات زیادی از یک بانک اطلاعاتی یا یک فایل برروی Server دریافت‌کنید، یا اطلاعات کاربر را به یک Server بفرستید، باید یک فرم HTML طراحی‌کرده و داده‌ها را به‌روش GET یا POST برای Server ارسال‌کنید. کاربر باید برروی دکمه Submit کلیک‌کند تا اطلاعات را ارسال‌کند، سپس منتظر پاسخ Server بماند، بعد یک صفحه جدید بارگذاری‌شده و نتایج را نشان می‌دهد.

این‌امر بدین‌دلیل رخ می‌دهد که Server یک صفحه جدید در هربار ارسال اطلاعات کاربر توسط فشردن دکمه Submit ایجاد می‌کند. لذا اینگونه برنامه‌ها، معمولاً کندتر و کمتر با کاربر دوست هستند.

به‌کمک AJAX، کد JavaScript شما می‌تواند مستقیماً با Server ارتباط برقرارکند، و یک‌پاسخ را از Server دریافت‌نماید - بدون آنکه نیاز به بارگذاری مجدد صفحه باشد. کاربر در همان صفحه باقی می‌ماند، و بدون اینکه متوجه درخواست‌ها و تبادل اطلاعات در پس‌زمینه شود، بخشی از صفحه که اطلاعات آن تغییرکرده‌است - نه کل صفحه، به‌روزرسانی می‌شود.

شئ XMLHttpRequest

با استفاده از شئ XMLHttpRequest، یک طراح Web می‌تواند یک صفحه را با داده‌هایی که از Server بعد از بارگذاری صفحه! دریافت کرده‌است، به‌روزرسانی‌کند.

AJAX در سال ۲۰۰۵ به پیشنهاد و توسط Google، جهانی شد :

پیشنهاد Google، استفاده از شئ XMLHttpRequest برای ساختن یک رابط تحت Web بسیار پویا بود: وقتی که شما در کادر جستجوی Google شروع به تایپ‌کردن کنید، یک کد JavaScript، حروف تایپ‌شده را به یک Server می‌فرستد و Server، فهرستی از موارد پیشنهادی را بازمی‌گرداند.

شئ XMLHttpRequest در Internet Explorer از نسخه ۵ به‌بعد، Safari نسخه ۱٫۲، Mozilla نسخه ۱ و FireFox (نسخه ۱٫۵)، Opera از نسخه ۸ به‌بعد و NetScape نسخه ۷ پشتیبانی می‌شود.

اولین برنامه AJAX :

برای درک اینکه AJAX چگونه کار می‌کند، ما یک برنامه AJAX کوچک ایجاد می‌کنیم. ابتدا، یک فرم استاندارد HTML طراحی می‌کنیم که دارای دو کادر متن است: UserName و Time. کادر متن UserName توسط کاربر و کادر متن Time توسط AJAX پر خواهد شد.

نام فایل HTML را testAjax.htm قرارداده و کدهای زیر را درداخل آن می‌نویسیم (دقت‌کنید که فرم HTML زیر، دکمه Submit ندارد) :

```
<html>
<body>
<form name="myForm">
Name: <input type="text" name="username" />
Time: <input type="text" name="time" />
</form>
</body>
</html>
```

سنگ بنای AJAX :

کلید اصلی و درواقع، سنگ بنای AJAX، شئ XMLHttpRequest است. مرورگرهای مختلف، روش‌های متفاوتی برای ایجاد شئ XMLHttpRequest دارند. Internet Explorer از یک شئ ActiveX استفاده می‌کند، درحالی‌که دیگر مرورگرها در شئ JavaScript داخلی (توکار) خود، به‌نام XMLHttpRequest بهره می‌برند.

برای ایجاد این شئ، و کارکردن با مرورگرهای مختلف، ما از یک دستور try و catch استفاده می‌کنیم. توضیحات بیشتر این دستور، در زبان JAVA موجود است. دراینجا به همین‌مقدار بسنده می‌کنیم که یک روش را امتحان می‌کند، اگر منجر به خطا شد، روش دیگر و همین‌طور الی آخر.

اجازه‌دهید فایل testAjax.htm خود را با این کد JavaScript که شئ XMLHttpRequest را ایجاد می‌کند، تکمیل کنیم :

```
<script type="text/javascript">
function ajaxFunction()
 {
 var xmlHttp;
 try
  {
  // Firefox, Opera ۸.۰+, Safari
  xmlHttp=new XMLHttpRequest();
  }
 catch (e)
  {
  // Internet Explorer
  try
   {
   xmlHttp=new ActiveXObject("Msxml۲.XMLHTTP");
   }
  catch (e)
   {
   try
    {
    xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
   catch (e)
    {
    alert("Your browser does not support AJAX!");
    return false;
```

```
        }
      }
    }
  }
</script>
```

این دستورات را به بعد از خط <body> اضافه می‌کنیم.

تشریح کد :

خط اول، یک متغیر به‌نام xmlHttp برای نگه‌داری شئ XMLHttpRequest ایجاد می‌کند. سپس تلاش می‌کند تا یک شئ با دستور new XMLHttpRequest() ایجاد نماید. این‌دستور درمورد مرورگرهای غیر Internet Explorer کار می‌کند. اگر این‌دستور منجر به شکست شود، از دستور new ActiveXObject("Msxml۲.XMLHTTP") که ویژه مرورگرهای نسخه ۶ به‌بعد است استفاده می‌کند. اگر مجدداً شکست‌خورد، از دستور new ActiveXObject("Microsoft.XMLHTTP") که ویژه مرورگرهای قبل از نسخه ۶ و بعداز نسخه ۵ است، استفاده می‌کند.

اگر هر سه روش فوق شکست‌خوردند، به‌معنی استفاده کاربر از یک مرورگر خیلی قدیمی است که از AJAX پشتیبانی نمی‌کند. لذا پیغام خطای مناسب برای کاربر تولید می‌شود.

نکته : کد مشخص‌کننده مرورگر فوق، طولانی و کمی پیچیده است. هرچند، این‌کد را می‌توانید هرگاه نیازمند ایجاد یک شئ XMLHttpRequest بودید، مورداستفاده قراردهید. لذا کافی‌است آنرا Copy و درجای موردنظر، Paste نمایید. کد فوق، با تمامی مرورگرهای محبوب ازجمله Internet Explorer، Opera، FireFox و Safari سازگار است.

خصوصیت onreadystatechange :

قبل از اینکه داده‌ها را به‌سمت Server ارسال کنیم، باید سه خصوصیت مهم شئ XMLHttpRequest را توضیح‌دهیم. یکی‌از این خصوصیت‌ها، onreadystatechange است. بعداز درخواست از Server، به تابعی نیازداریم که بتواند داده‌هایی را که ازسمت Server بازگردانده می‌شوند، دریافت‌کند. خصوصیت

onreadystatechange محل نگهداری تابعی است که پاسخ Server را پردازش می‌کند. کد زیر، یک تابع خالی و همچنین خصوصیت onreadystatechange را به‌صورت هم‌زمان تعریف می‌کند :

```
xmlHttp.onreadystatechange=function()
  {
    // We are going to write some code here
  }
```

خصوصیت readyState :

این خاصیت، وضعیت پاسخ Server را نگه‌داری می‌کند. هرزمان که readyState تغییرکند، تابع onreadystatechange فراخوانی و اجرا می‌شود. دراینجا، مقادیری که خصوصیت readyState قبول می‌کند را مشاهده می‌کنید :

وضعیت	شرح
۰	درخواست تنظیم نشده‌است.
۱	درخواست تنظیم شده‌است.
۲	درخواست ارسال شده‌است.
۳	درخواست درحال پردازش است.
۴	درخواست تکمیل شده‌است.

ما می‌خواهیم یک دستور If به تابع onreadystatechange اضافه‌کنیم تا بررسی‌کند که آیا درخواست ما تکمیل شده‌است یا نه (این بدان‌معنی است که می‌توانیم داده‌های خود را دریافت‌کنیم یا خیر)؟ به دستورات مربوطه دقت‌کنید :

```
xmlHttp.onreadystatechange=function()
{
if(xmlHttp.readyState==۴)
{
document.myForm.time.value=xmlHttp.responseText;
}
}
```

ارسال درخواست به Server :

برای ارسال یک درخواست به Server، ما از متدهای open() و send() استفاده می‌کنیم. متد open() سه پارامتر ورودی دارد. اولین پارامتر، متد مورداستفاده برای ارسال درخواست را مشخص می‌کند (GET یا POST). دومین پارامتر بیانگر آدرس Script (کد) موردنظر برروی کامپیوتر Server است. سومین پارامتر مشخص کننده این‌امر است که درخواست باید به‌صورت غیرهم‌زمان پردازش شود (true) یا هم‌زمان (false). متد send()، درخواست تنظیم‌شده را به Server ارسال می‌کند. اگر فرض‌کنیم که فایل HTML و ASP موردنظر ما هردو در یک پوشه مشترک قراردارند، کد موردنظر چیزی شبیه به این کد خواهد بود :

```
xmlHttp.open("GET","time.asp",true);
xmlHttp.send(null);
```

حال باید تصمیم‌بگیریم که تابع AJAX ما چه‌زمانی باید اجرا شود. در ادامه، ما به تابع اجازه‌می‌دهیم تا در «پشت صحنه»، وقتی‌که کاربر کلیدی را در کادر متن UserName فشاردهد (یا نگه‌دارد)، اجرا شود. لذا کد ایجادکننده کادر متن UserName را به‌صورت زیر تغییر می‌دهیم :

```
Name: <input type="text" onkeydown="ajaxFunction();" name="username" />
```

کد ASP سمت Server :

حال باید دستوری ایجاد کنیم که زمان جاری Server را نشان‌دهد. خصوصیت responseText (که قبلاً توضیح دادیم)، داده‌هایی را که ازسمت Server بازگردانده می‌شود، ذخیره می‌کند. دراینجا ما زمان جاری را بازمی‌گردانیم. کد زیر را در فایل time.asp می‌نویسیم :

```
<%
response.expires=-۱
response.write(time)
%>
```

نکته : خصوصیت Expires مدت‌زمانی را (برحسب دقیقه) مشخص می‌کند که یک صفحه، برروی مرورگر باید قبل‌از اینکه ازبین برود، ذخیره‌گردد. اگر کاربر همان صفحه را قبل‌از اینکه ازبین برود، بازکند، درآن‌صورت، نسخه ذخیره‌شده بازخواهدشد و صفحه جدید از Server درخواست نمی‌شود. دستور Response.Expires=-۱ بیان‌گر این‌مطلب است که صفحه، هرگز ذخیره نمی‌شود و همیشه باید از Server درخواست‌گردد (بدیهی‌است که اگر صفحه ذخیره‌گردد، درآن‌صورت، زمان قبلی که در صفحه قبلی ذخیره شده‌است، نشان‌داده خواهدشد، نه زمان جدید).

یک مثال عملی تر :

در این مثال، صفحه ای طراحی خواهیم کرد که با فشاردادن هر حرف، فهرستی از اسامی موجود در فایل اسامی را که محتوای کادر متن در آنها وجود دارد، نشان دهد و اگر کادر متن خالی باشد، چیزی نشان ندهد.

```
<html>
<head>
<script type="text/javascript">
function showHint(str)
{
if (str.length==۰)
  {
  document.getElementById("txtHint").innerHTML="";
  return;
  }
xmlHttp=GetXmlHttpObject()
if (xmlHttp==null)
  {
  alert ("Your browser does not support AJAX!");
  return;
  }
var url="gethint.asp";
url=url+"?q="+str;
url=url+"&sid="+Math.random();
xmlHttp.onreadystatechange=stateChanged;
```

```
xmlHttp.open("GET",url,true);
xmlHttp.send(null);
}
</script>
</head>
<body>
<form>
First Name: <input type="text" id="txt۱" onkeyup="showHint(this.value)">
</form>
<p>Suggestions: <span id="txtHint"></span></p>
</body>
</html>
```

کمی توضیح :

همان‌طور که می‌بینید، این صرفاً یک فرم HTML ساده با یک کادر متن به‌نام txt۱ است. یک رویداد برای کادر متن تعریف می‌شود که درزمان رهاکردن یک کلید درداخل کادر متن، فراخوانی و اجرا می‌شود. دستور بعد، حاوی یک Span (محدوده) به‌نام txtHint است. Span به‌عنوان یک نگهدارنده برای داده‌هایی که ازسمت Server دریافت خواهیم کرد، عمل خواهد نمود. وقتی که کاربر داده‌ها را وارد می‌کند، یک تابع به‌نام showHint() اجرا می‌شود. اجرای این تابع براساس رویداد onkeyup است. به‌بیان‌دیگر، هربار که کاربر انگشت خود را ازروی یک کلید از صفحه‌کلید بردارد، درحالی‌که کادر متن فعال‌باشد، تابع showHint فراخوانی می‌شود.

تابع، هربار که یک کارکتر وارد کادر متن می‌شود، اجرا خواهد شد. اگر چیزی درداخل کادر متن باشد (str.length > ۰)، تابع کارهای زیر را انجام می‌دهد :

۱- آدرس نام فایل حاوی اسامی را برای ارسال به Server تنظیم می‌کند.

۲- عبارت =q? را به‌همراه محتوای کادر متن به آدرس اضافه می‌کند (شبیه‌سازی متد GET).

۳- عبارت =&sid را به‌همراه یک عدد تصادفی به آدرس اضافه می‌کند تا یک آدرس ذخیره‌شده، بازنشود و هربار، درخواست جدیدی تولید شود.

۴- شئ XMLHTTP را طوری تنظیم می‌کند که هرزمان خصوصیت onreadystatechanged تغییریافت، تابع stateChanged فراخوانی و اجرا شود.

۵- شئ XMLHTTP را با آدرس تولیدشده، باز می‌کند.

۶- درخواست HTTP را برای Server ارسال می‌کند.

اگر کادر متن خالی‌باشد، تابع مربوطه، به‌سادگی محتوای txtHint را پاک می‌کند.

تشریح کد - تابع GetXmlHttpObject() :

مثال فوق، تابعی را تحت‌عنوان GetXmlHttpObject() فراخوانی می‌کند. هدف این تابع، رفع مشکل ایجاد اشیاء مختلف XMLHTTP برای مرورگرهای مختلف است. درواقع چیز جدیدی نیست. همان دستورات ایجاد شئ XMLHttpRequest است که دراینجا، تحت‌عنوان یک مجزا، نام‌گذاری شده‌است :

```
function GetXmlHttpObject()
{
  var xmlHttp=null;
  try
    {
   // Firefox, Opera ۸.۰+, Safari
   xmlHttp=new XMLHttpRequest();
```

```
    }
  catch (e)
    {
    // Internet Explorer
    try
      {
      xmlHttp=new ActiveXObject("Msxml۲.XMLHTTP");
      }
    catch (e)
      {
      xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
      }
    }
  return xmlHttp;
}
```

تشریح کد - تابع stateChanged() :

تابع stateChanged() حاوی کد زیر است :

```
function stateChanged()
  {
    if (xmlHttp.readyState==۴)
```

```
        {
document.getElementById("txtHint").innerHTML=xmlHttp.responseText;
        }
  }
```

تابع stateChanged() هربار که وضعیت شئ XMLHTTP تغییرکند، فراخوانی می‌شود. وقتی‌که وضعیت به حالت ۴ ("تکمیل‌شده") تغییرکند، محتوای نگهدارنده txtHint با متن پاسخ پر می‌شود.

کد AJAX مربوط به فایل HTML :

در اینجا، کد فایل HTML را که می‌توانید با نام دلخواه ذخیره‌کنید، مشاهده می‌نمایید :

```
<html>
<head>
<script src="clienthint.js">
</script>
</head>
<body>
<form>
First Name:<input type="text" id="txt۱" onkeyup="showHint(this.value)">
</form>
<p>Suggestions: <span id="txtHint"></span></p>
```

```
</body>
</html>
```

کد AJAX مربوط به فایل JavaScript :

در اینجا، کد فایل clienthint.js را مشاهده می‌کنید که درقسمت Head فایل HTML، فراخوانی می‌شود :

```
var xmlHttp
function showHint(str)
{
  if (str.length==۰)
  {
    document.getElementById("txtHint").innerHTML="";
    return;
  }
  xmlHttp=GetXmlHttpObject()
  if (xmlHttp==null)
  {
    alert ("Your browser does not support AJAX!");
    return;
  }
  var url="gethint.asp";
```

```
  url=url+"?q="+str;
  url=url+"&sid="+Math.random();
  xmlHttp.onreadystatechange=stateChanged;
  xmlHttp.open("GET",url,true);
  xmlHttp.send(null);
}
function stateChanged()
{
  if (xmlHttp.readyState==۴)
  {
     document.getElementById("txtHint").innerHTML=xmlHttp.responseText;
  }
}



function GetXmlHttpObject()
{
  var xmlHttp=null;
  try
  {
    // Firefox, Opera ۸.۰+, Safari
```

```
    xmlHttp=new XMLHttpRequest();
  }
  catch (e)
  {
    // Internet Explorer
    try
    {
      xmlHttp=new ActiveXObject("Msxml۲.XMLHTTP");
    }
    catch (e)
    {
      xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
  return xmlHttp;
}
```

صفحه Server **تکنیک** AJAX - ASP **و** PHP :

هیچ چیزی درمورد AJAX تحت‌عنوان Server وجود ندارد. صفحات AJAX را می‌توان با هر Server اینترنت دلخواه، پیاده‌سازی نمود.

صفحه Server فراخوانی‌شده توسط JavaScript در مثال قبل، یک فایل ASP ساده به‌نام gethint.asp بود.

در زیر، ما دو مثال از کد صفحه Server با زبان‌های ASP و PHP ارائه می‌کنیم که می‌توانید از هرکدام که مایل‌باشید، به‌عنوان فایل حاوی اسامی، در مثال فوق استفاده‌کنید.

مثال ASP تکنیک AJAX :

در اینجا، کد فایل gethint.asp را مشاهده می‌کنید که در فایل clienthint.js، فراخوانی می‌شود :

```
'get the q parameter from URL
q=ucase(request.querystring("q"))
'lookup all hints from array if length of q>۰
if len(q)>۰ then  hint=""
  for i=۱ to ۳۰
    if q=ucase(mid(a(i),۱,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
%>



<%
response.expires=-١
dim a(٣٠)'Fill up array with names
a(١)="Anna"
a(٢)="Brittany"
a(٣)="Cinderella"
a(٤)="Diana"
a(٥)="Eva"
a(٦)="Fiona"
a(٧)="Gunda"
a(٨)="Hege"
a(٩)="Inga"
a(١٠)="Johanna"
a(١١)="Kitty"
a(١٢)="Linda"
a(١٣)="Nina"
```

a(۱۴)="Ophelia"

a(۱۵)="Petunia"

a(۱۶)="Amanda"

a(۱۷)="Raquel"

a(۱۸)="Cindy"

a(۱۹)="Doris"

a(۲۰)="Eve"

a(۲۱)="Evita"

a(۲۲)="Sunniva"

a(۲۳)="Tove"

a(۲۴)="Unni"

a(۲۵)="Violet"

a(۲۶)="Liza"

a(۲۷)="Elizabeth"

a(۲۸)="Ellen"

a(۲۹)="Wenche"

a(۳۰)="Vicky"

مثال PHP تکنیک AJAX :

در اینجا، کد فایل gethint.php را مشاهده می‌کنید که در فایل clienthint.js، با تغییر gethint.asp به gethint.php، فراخوانی می‌شود :

```
//get the q parameter from URL
$q=$_GET["q"];
//lookup all hints from array if length of q>۰
if (strlen($q) > ۰)
{
  $hint="";
  for($i=۰; $i<count($a); $i++)
  {
  if (strtolower($q)==strtolower(substr($a[$i],۰,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;
?>
```

```
<?php
header("Cache-Control: no-cache, must-revalidate");
 // Date in the past
header("Expires: Mon, ٢٦ Jul ١٩٩٧ ٠٥:٠٠:٠٠ GMT");
// 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";
```

AJAX و بانک‌های اطلاعاتی:

در مثال AJAX زیر، به شما نشان می‌دهیم که چگونه یک صفحه Web می‌تواند اطلاعات را از بانک اطلاعاتی بااستفاده‌از تکنولوژی AJAX استخراج کند. یک فایل HTML با نام دلخواه و با کد زیر ایجاد می‌کنیم :

```
<html>
<head>
<script src="selectcustomer.js"></script>
</head>
<body>
<form>
Select a Customer:
<select name="customers" onchange="showCustomer(this.value)">
<option value="ALFKI">Alfreds Futterkiste
<option value="NORTS ">North/South
<option value="WOLZA">Wolski Zajazd
</select>
</form>
<p>
```

```
<div id="txtHint"><b>Customer info will be listed here.</b></div>
</p>
</body>
</html>
```

توضیح :

همان‌طور که می‌توانید مشاهده‌کنید، این فایل، یک فرم HTML ساده با یک لیست پایین‌افتادنی به‌نام customers است. در ادامه، یک تگ div به‌نام txtHint قراردارد که برای نگهداری اطلاعات دریافت‌شده از Web Server به‌کار می‌رود.

وقتی که کاربر چیزی را انتخاب‌کند، تابع showCustomer() توسط رویداد onchange فراخوانی می‌شود و محتوای لیست customers برای آن ارسال می‌گردد. به‌بیان ساده‌تر، هربار که کاربر مقدار لیست customers را تغییردهد، تابع showCustomer فراخوانی‌می‌شود.

کد JavaScript مربوطه (selectcustomer.js) را در ادامه می‌بینید.

AJAX JavaScript :

```
var xmlHttp
function showCustomer(str)
{
xmlHttp=GetXmlHttpObject();
if (xmlHttp==null)
  {
  alert ("Your browser does not support AJAX!");
  return;
```

```
  }
var url="getcustomer.asp";
url=url+"?q="+str;
url=url+"&sid="+Math.random();
xmlHttp.onreadystatechange=stateChanged;
xmlHttp.open("GET",url,true);
xmlHttp.send(null);
}
function stateChanged()
{
if (xmlHttp.readyState==۴)
{
document.getElementById("txtHint").innerHTML=xmlHttp.responseText;
}
}



function GetXmlHttpObject()
{
var xmlHttp=null;
try
  {
  // Firefox, Opera ۸.۰+, Safari
  xmlHttp=new XMLHttpRequest();
```

```
  }
catch (e)
  {
  // Internet Explorer
  try
    {
    xmlHttp=new ActiveXObject("Msxml۲.XMLHTTP");
    }
  catch (e)
    {
    xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
return xmlHttp;
}
```

صفحه Server **مربوطبه** AJAX :

صفحه Server مربوطه که توسط کد JavaScript فراخوانی می‌شود، یک فایل ASP ساده به‌نام getcustomer.asp است. مجدداً یادآور می‌شویم که می‌توان به‌راحتی از فایل‌های PHP و دستورات آن‌زبان استفاده‌نمود و AJAX محدود به ASP یا هرگونه زبان Server دیگری نیست (northwind.db یک بانک اطلاعاتی نمونه است که همراه با نرم افزار Microsoft Access نصب می‌شود) .

کد فایل getcustomer.asp به‌صورت زیر است :

```
<%
```

```
response.expires=-۱
sql="SELECT * FROM CUSTOMERS WHERE CUSTOMERID="
sql=sql & "'" & request.querystring("q") & "'"
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.۴.۰"
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 و تعامل با XML :

AJAX می‌تواند برای ارتباط فعل‌وانفعالی و پویا با یک فایل XML نیز مورداستفاده قرارگیرد. در مثال زیر، نشان می‌دهیم که چگونه می‌توان از یک صفحه Web برای استخراج اطلاعات از یک فایل XML به‌کمک تکنولوژی AJAX استفاده‌نمود. فایل HTML زیر را با نام دلخواه ایجاد کنید :

```
<html>
<head>
<script src="selectcd.js"></script>
</head>
<body>
<form>
Select a CD:
<select name="cds" onchange="showCD(this.value)">
<option value="Bob Dylan">Bob Dylan</option>
<option value="Bonnie Tyler">Bonnie Tyler</option>
<option value="Dolly Parton">Dolly Parton</option>
</select>
</form>
<p>
<div id="txtHint"><b>CD info will be listed here.</b></div>
</p>
</body>
</html>
```

توضیح :

همان‌طور که می‌توانید مشاهده‌کنید، این فایل، یک فرم HTML ساده با یک لیست پایین‌افتادنی به‌نام cds است. در ادامه، یک تگ div به‌نام txtHint قراردارد که برای نگهداری اطلاعات دریافت‌شده از Web Server به‌کار می‌رود.

وقتی که کاربر چیزی را انتخاب‌کند، تابع ()showCD توسط رویداد onchange فراخوانی می‌شود و محتوای لیست cds برای آن ارسال می‌گردد. به‌بیان ساده‌تر، هربار که کاربر مقدار لیست cds را تغییردهد، تابع showCD فراخوانی‌می‌شود.

کد JavaScript مربوطه (selectcd.js) را در ادامه می‌بینید.

AJAX JavaScript :

در اینجا، کد فایل selectcd.js را مشاهده می‌کنید که درقسمت Head فایل HTML، فراخوانی می‌شود :

```
var xmlHttp
function showCD(str)
{
xmlHttp=GetXmlHttpObject();
if (xmlHttp==null)
  {
  alert ("Your browser does not support AJAX!");
  return;
  }
var url="getcd.asp";
url=url+"?q="+str;
url=url+"&sid="+Math.random();
xmlHttp.onreadystatechange=stateChanged;
xmlHttp.open("GET",url,true);
xmlHttp.send(null);
}
```

```
function stateChanged()
{
if (xmlHttp.readyState==۴)
{
document.getElementById("txtHint").innerHTML=xmlHttp.responseText;
}
}



function GetXmlHttpObject()
{
var xmlHttp=null;
try
  {
  // Firefox, Opera ۸.۰+, Safari
  xmlHttp=new XMLHttpRequest();
  }
catch (e)
  {
  // Internet Explorer
  try
    {
    xmlHttp=new ActiveXObject("Msxml۲.XMLHTTP");
```

```
    }
  catch (e)
    {
    xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
return xmlHttp;
}
```

صفحه Server **مربوطبه** AJAX :

صفحه Server مربوطه که توسط کد JavaScript فراخوانی می‌شود، یک فایل ASP ساده به‌نام getcd.asp است. باز هم یادآور می‌شویم که می‌توان به‌راحتی از فایل‌های PHP و دستورات آن‌زبان استفاده‌نمود و AJAX محدود به ASP یا هرگونه زبان Server دیگری نیست (cd_catalog.xml یک فایل XML نمونه است که می‌توانید از آدرس http://www.w۳schools.com/ajax/cd_catalog.xml آنرا دریافت‌کنید) .

کد فایل getcd.asp به‌صورت زیر است :

```
<%
response.expires=-۱
q=request.querystring("q")
set xmlDoc=Server.CreateObject("Microsoft.XMLDOM")
xmlDoc.async="false"
xmlDoc.load(Server.MapPath("cd_catalog.xml"))
set nodes=xmlDoc.selectNodes("CATALOG/CD[ARTIST='" & q & "']")
```

```
for each x in nodes
  for each y in x.childnodes
    response.write("<b>" & y.nodename & ":</b> ")
    response.write(y.text)
    response.write("<br />")
  next
next
%>
```

ResponseXML :

درحالی‌که responseText، پاسخ HTTP را به‌صورت یک رشته متنی بر می‌گرداند، دستور responseXML، پاسخ را به‌صورت XML باز می‌گرداند.

خصوصیت ResponseXML یک شئ سند XML باز می‌گرداند که می‌تواند توسط خاصیت‌ها و متدهای درخت گره W۳C DOM مورد بررسی و پردازش قرارگیرد.

در مثال بعدی AJAX، از یک صفحه Web برای استخراج اطلاعات از یک بانک اطلاعاتی استفاده می‌کنیم، اما این‌بار، داده‌های استخراج‌شده از بانک اطلاعاتی، به‌صورت یک سند XML تبدیل‌شده و سپس، از DOM برای استخراج مقادیر جهت نمایش استفاده می‌کنیم.

فایل HTML :

یک فایل HTML با نام دلخواه و با کد زیر ایجاد می‌کنیم :

```
<html>
```

```
<head>
<script src="selectcustomer_xml.js"></script>
</head>
<body>
<form action="">
Select a Customer:
<select name="customers" onchange="showCustomer(this.value)">
<option value="ALFKI">Alfreds Futterkiste</option>
<option value="NORTS ">North/South</option>
<option value="WOLZA">Wolski Zajazd</option>
</select>
</form>
<b><span id="companyname"></span></b><br/>
<span id="contactname"></span><br/>
<span id="address"></span>
<span id="city"></span><br/>
<span id="country"></span>
</body>
</html>
```

توضیح :

همان‌طور که می‌توانید مشاهده‌کنید، این فایل، یک فرم HTML ساده با یک لیست پایین‌افتادنی به‌نام customers است. در ادامه، یک تگ div به‌نام txtHint قراردارد که برای نگهداری اطلاعات دریافت‌شده از Web Server به‌کار می‌رود.

وقتی که کاربر چیزی را انتخاب‌کند، تابع showCustomer() توسط رویداد onchange فراخوانی می‌شود و محتوای لیست customers برای آن ارسال می‌گردد. به‌بیان ساده‌تر، هربار که کاربر مقدار لیست customers را تغییردهد، تابع showCustomer فراخوانی‌می‌شود.

کد JavaScript مربوطه (selectcustomer_xml.js) را در ادامه می‌بینید.

AJAX JavaScript :

در اینجا، کد فایل selectcustomer_xml.js را مشاهده می‌کنید که درقسمت Head فایل HTML، فراخوانی می‌شود :

```
var xmlHttp
function showCustomer(str)
{
xmlHttp=GetXmlHttpObject();
if (xmlHttp==null)
  {
  alert ("Your browser does not support AJAX!");
  return;
  }
var url="getcustomer_xml.asp";
url=url+"?q="+str;
url=url+"&sid="+Math.random();
xmlHttp.onreadystatechange=stateChanged;
xmlHttp.open("GET",url,true);
xmlHttp.send(null);
```

```
}
function stateChanged()
{
if (xmlHttp.readyState==۴)
{
var xmlDoc=xmlHttp.responseXML.documentElementa;
document.getElementById("companyname").innerHTML=
xmlDoc.getElementsByTagName("compname")[۰].childNodes[۰].nodeValue;
document.getElementById("contactname").innerHTML=
xmlDoc.getElementsByTagName("contname")[۰].childNodes[۰].nodeValue;
document.getElementById("address").innerHTML=
xmlDoc.getElementsByTagName("address")[۰].childNodes[۰].nodeValue;
document.getElementById("city").innerHTML=
xmlDoc.getElementsByTagName("city")[۰].childNodes[۰].nodeValue;
document.getElementById("country").innerHTML=
xmlDoc.getElementsByTagName("country")[۰].childNodes[۰].nodeValue;
}
}



function GetXmlHttpObject()
{
```

```
var xmlHttp=null;
try
  {
  // Firefox, Opera ۸.۰+, Safari
  xmlHttp=new XMLHttpRequest();
  }
catch (e)
  {
  // Internet Explorer
  try
    {
    xmlHttp=new ActiveXObject("Msxml۲.XMLHTTP");
    }
  catch (e)
    {
    xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
return xmlHttp;
}
```

صفحه Server مربوطبه AJAX :

توابع showCurtomer() و GetXmlHttpObject() دقیقاً مشابه مثال‌های قبل هستند. تابع stateChanged() نیز به‌روشی مشابه در مثال‌های قبل به‌کارگرفته شده‌است. لیکن این‌بار، تعداد Spanها بیشترشده و برای هرکدام، یک‌دستور ذکر می‌شود. در ادامه، این‌بار نتیجه را به‌عنوان یک سند XML (توسط responseXML) بازگردانده و از DOM برای استخراج مقادیری که می‌خواهیم نشان‌دهیم، استفاده خواهیم‌کرد.

```
<%
response.expires=-۱
response.contenttype="text/xml"
sql="SELECT * FROM CUSTOMERS "
sql=sql & " WHERE CUSTOMERID='" & request.querystring("q") & "'"
on error resume next
set conn=Server.CreateObject("ADODB.Connection")
conn.Provider="Microsoft.Jet.OLEDB.۴.۰"
conn.Open(Server.Mappath("/db/northwind.mdb"))
set rs=Server.CreateObject("ADODB.recordset")
rs.Open sql, conn
if err <> ۰ then
response.write(err.description)
set rs=nothing
set conn=nothing
else
response.write("<?xml version='۱.۰' encoding='ISO-۸۸۵۹-۱'?>")
response.write("<company>")
response.write("<compname>" &rs.fields("companyname")& "</compname>")
```

```
response.write("<contname>" &rs.fields("contactname")& "</contname>")
response.write("<address>" &rs.fields("address")& "</address>")
response.write("<city>" &rs.fields("city")& "</city>")
response.write("<country>" &rs.fields("country")& "</country>")
response.write("</company>")
end if
on error goto ۰
%>
```

توضیح :

به خط دوم در کد ASP فوق دقت‌کنید : response.contenttype="text/xml"

خصوصیت ContentType، نوع محتویات HTTP را درمورد شئ پاسخ (Response) مشخص‌می‌کند. نوع پیش‌فرض این خصوصیت، "text/html" است. این‌بار ما می‌خواهیم که نوع محتوا XML باشد. لذا از "text/xml" استفاده می‌کنیم.

سپس داده‌ها را از بانک اطلاعاتی انتخاب‌کرده، و یک سند XML با این داده‌ها ایجادمی‌کنیم.