AJAX Tutorial

AJAX está para Asynchronous JavaScript and XML.

AJAX é um tipo de programação feito popular em 2005 por Google (com Google sugerir).

AJAX não é uma língua de programação nova, mas uma maneira nova usar padrões existentes.

Com o AJAX você pode criar melhor, rapidamente, e aplicações mais user-friendly da web applications.

AJAX é baseado em pedidos do Javascript e do HTTP.

Introdução de AJAX

AJAX está para Asynchronous JavaScript and XML.

O que você deve já saber

Antes que você continue você deve ter uma compreensão básica do seguinte:

- HTML / XHTML
- JavaScript

AJAX = Asynchronous JavaScript and XML

AJAX não é uma língua de programação nova, mas uma técnica para criar melhor e mais rapidamente, e uma aplicação mais interativa da web applications.

Com o AJAX, seu Javascript pode comunicar-se diretamente com o usuário, usando o objeto de **XMLHttpRequest** do Javascript. Com este objeto, seu Javascript pode negociar dados com um web server, sem recarregar a página.

AJAX usa transferência de dados assíncrona (pedidos do HTTP) entre o browser e o web server, permitindo que os Web pages peçam bocados de informação pequenos do usuário em vez das páginas inteiras.

A técnica de AJAX faz aplicações da Internet menores, mais rapidamente e mais user-friendly.

AJAX é um independente da tecnologia do browser do software da web server.

AJAX é baseado em padrões Web Standards

AJAX é baseado nos seguintes padrões da web Standards:

- JavaScript

- XML
- HTML
- CSS

Os padrões da web Standards usados em AJAX são definidos bem, e suportados por todos os principais browsers. As aplicações de AJAX são browser e independente da plataforma.

AJAX é sobre aplicações melhores da Internet

As aplicações da Web têm muitos benefícios sobre aplicações desktop; podem alcançar uma audiência maior, é mais fácil de instalar e suportar, e mais fácil de tornar-se.

Entretanto, as aplicações da Internet não são sempre como “ricas” e user-friendly como aplicações desktop tradicionais.

Com AJAX, as aplicações da Internet podem ser feitas mais ricas e mais user-friendly.

Você pode começar usar AJAX hoje

Não há nada novo aprender.

AJAX é baseado em padrões existentes. Estes padrões foram usados por a maioria de colaboradores por diversos anos.

AJAX Http Requests

AJAX usa pedidos do HTTP

No coding tradicional do Javascript, se você quiser começar alguma informação de uma base de dados ou de um arquivo no usuário, ou emitir a informação do usuário a um usuário, você terá que fazer um HTML dar forma e COMEÇAR ou AFIXAR os dados ao usuário. O usuário terá que enviar “submete” a tecla para emitir/começa a informação, espera para que o usuário responda, então uma página nova carregará com os resultados.

Porque o usuário retorna uma página nova cada vez o usuário submete a entrada, as aplicações tradicionais da web application podem funcionar lentamente e tender a ser mais menos user-friendly.

Com o AJAX, seu Javascript comunica-se diretamente com o usuário, através do objeto de **XMLHttpRequest** do Javascript

Com um pedido do HTTP, uma página web pode fazer um pedido, e começa uma resposta de um web server - sem recarregar a página. O usuário permanecerá na mesma página, e ou não observarão que os scripts pedem páginas, ou emitem dados a um usuário no fundo.

O objeto de XMLHttpRequest

Usando o objeto de XMLHttpRequest, um colaborador da web standart pode atualizar uma página com dados do usuário depois que a página carregou!

AJAX foi feito popular em 2005 pela Google (com Google suggest(Sugerir)).

Google Suggest está usando o objeto de **XMLHttpRequest** criar uma relação muito dinâmica da web interface: Quando você começa datilografar na caixa da busca do Google, um Javascript emite as letras fora a um usuário e o usuário retorna uma lista das sugestões.

O objeto de **XMLHttpRequest** é suportado no Internet Explorer 5.0+, no Safari 1.2, no Mozilla 1.0/Firefox, ópera 8+, e no Netscape 7.

Exemplo de AJAX

Sua primeira aplicação em AJAX

Para compreender como AJAX trabalha, nós criaremos uma aplicação pequena de AJAX.

Primeiramente, nós estamos indo criar um formulário padrão do HTML com os dois campos do texto: username e time. O campo do username será preenchido pelo usuário e o campo do time será preenchido dentro usando AJAX.

O arquivo do HTML será nomeado “testAjax.htm”, e olha como esta (observação que o formulário do HTML abaixo tem nenhum submeter à tecla!):

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

Os capítulos seguintes explicarão as chaves de AJAX.

Support do Browser de AJAX

AJAX - Support do Browser

A chave de AJAX é o objeto de **XMLHttpRequest**.

Os browsers diferentes usam métodos diferentes para criar o objeto de **XMLHttpRequest**.

O Internet Explorer usa um **ActiveXObject**, quando outros browsers usarem o objeto interno do Javascript chamado **XMLHttpRequest**.

Para criar este objeto, e para tratar dos browsers diferentes, nós estamos indo usar uma "try ... catch" a indicação. Você pode ler mais sobre a tentativa e try ... catch statement em nosso tutorial do Javascript.

Vamos atualizar nosso arquivo de “testAjax.htm” com o Javascript que cría o objeto de **XMLHttpRequest**:

```
<html>
<body>
<script type="text/javascript">
function ajaxFunction()
  {
  var xmlHttp;
  try
    {
    // Firefox, Opera 8.0+, Safari
    xmlHttp=new XMLHttpRequest();
    }
  catch (e)
    {
    // Internet Explorer
    try
      {
      xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
      }
    catch (e)
      {
      try
        {
        xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
        }
      catch (e)
        {
        alert("Your browser does not support AJAX!");
        return false;
        }
      }
    }
  }
</script>
<form name="myForm">
Name: <input type="text" name="username" />
Time: <input type="text" name="time" />
</form>
</body>
</html>
```

Exemplo explicado:

Criar primeiramente um **xmlHttp** variável para prender o objeto de **XMLHttpRequest**.

Tentar então criar o objeto com **xmlHttp=new XMLHttpRequest()**. Isto é para os browsers de Firefox, de ópera, e de Safari. Se isso falhar, tentativa **xmlHttp=new ActiveXObject("Msxml2.XMLHTTP")** o qual é para o Internet Explorer 6.0+, se

aquele falhar também, tentativa **xmlHttp=new ActiveXObject("Microsoft.XMLHTTP")** o qual é para o Internet Explorer 5.5+

Se nenhuns dos três métodos trabalharem, o usuário tem um browser muito antigo, e começara um alerta indicando que o browser não suporta AJAX.

Nota: O código browser-específico acima é por muito tempo e completamente complexo. Entretanto, este é o código que você pode se usar cada vez que você necessita criar um objeto de **XMLHttpRequest**, assim que você pode apenas cópia e colá-lo sempre que você o necessita. O código acima é compatível com todos os browsers populares: Internet Explorer, ópera, Firefox, e Safari.

O capítulo seguinte mostra como usar o objeto de **XMLHttpRequest** comunicar-se com o usuário.

AJAX - O objeto de XMLHttpRequest

AJAX - Mais sobre o objeto de XMLHttpRequest

Antes de emitir dados ao usuário, nós temos que explicar três propriedades importantes do objeto de **XMLHttpRequest**.

A propriedade do onreadystatechange

Depois que um pedido ao usuário, nós necessitamos de uma função que possa receber os dados que são retornados pelo usuário.

A propriedade do **onreadystatechange** armazena a função que processará a resposta de um usuário. O seguinte código define uma função vazia e ajusta a propriedade do **onreadystatechange** ao mesmo tempo:

```
xmlHttp.onreadystatechange=function()
  {
  // nós estamos indo escrever aqui algum código
  }
```

A propriedade do readyState

A propriedade do **readyState** prende o status da resposta do usuário. Cada vez que o **readyState** muda, a função do **onreadystatechange** estará executada.

Estão aqui os valores possíveis para a propriedade do **readyState**:

Estado	Descrição
0	Os pedidos não são inicializados
1	Que o pedido foi ajustado acima
2	O pedido foi emitido
3	O pedido está no processo

4	O pedido está completo

Nós estamos indo adicionar se a indicação à função do **onreadystatechange** ao teste se nossa resposta estiver completa (isto significar que nós podemos começar nossos dados):

```
xmlHttp.onreadystatechange=function()
  {
  if(xmlHttp.readyState==4)
    {
    // Começar os dados da resposta do usuário
    }
  }
```

A propriedade do responseText

Os dados emitidos para trás do usuário podem ser recuperados com a propriedade do **responseText**.

Em nosso código, nós ajustaremos o valor de nosso campo da entrada do “time(tempo)” igual ao **responseText**:

```
xmlHttp.onreadystatechange=function()
  {
  if(xmlHttp.readyState==4)
    {
    document.myForm.time.value=xmlHttp.responseText;
    }
  }
```

O capítulo seguinte mostra como pedir o usuário para alguns dados!

AJAX - Pedir um usuário

AJAX - Emitindo um pedido ao usuário

Para emitir fora de um pedido ao usuário, nós usamos o método **open()** e o método **send()**.

O método **open()** faz um exame de três argumentos. O primeiro argumento define que método para se usar quando emitindo o pedido (**GET** ou **POST**). O segundo argumento especifica o URL do server-side script. O terceiro argumento especifica que o pedido deve ser segurado asynchronously. O método da **send()** emite o pedido fora ao usuário. Se nós supuséssemos que a lima do HTML e do ASP está no mesmo diretório, o código seria:

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

Agora nós devemos decidir-se quando a função de AJAX deve ser executada. Nós deixaremos a função funcionar “atrás das cenas” quando o usuário datilografa algo no campo do texto do username:

```
<form name="myForm">
Name: <input type="text"
onkeyup="ajaxFunction();" name="username" />
Time: <input type="text" name="time" />
</form>
```

Nossa lima AJAX-ready updated de “testAjax.htm” olha agora como está:

```
<html>
<body>
<script type="text/javascript">
function ajaxFunction(){
  var xmlHttp;
  try {
    // Firefox, Opera 8.0+, Safari
    xmlHttp=new XMLHttpRequest();
  } catch (e) {
    // Internet Explorer
    try {
      xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
    } catch (e){
      try {
        xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
      } catch (e) {
        alert("Your browser does not support AJAX!");
        return false;
      }
    }
  }
    xmlHttp.onreadystatechange=function() {
      if(xmlHttp.readyState==4) {
        document.myForm.time.value=xmlHttp.responseText;
      }
    }
    xmlHttp.open("GET","time.asp",true);
    xmlHttp.send(null);
}
</script>
<form name="myForm">
Name: <input type="text" onkeyup="ajaxFunction();" name="username" />
Time: <input type="text" name="time" />
</form>
</body>
</html>
```

O capítulo seguinte faz nossa aplicação de AJAX completa com o script de “time.asp”.

AJAX – O Script Server-Side

AJAX – O Script ASP Server-Side

Agora nós estamos indo criar o script que indica o tempo atual do usuário. A propriedade do **responseText** (explicada no capítulo precedente) armazenará os dados retornados do usuário. Aqui nós queremos emitir para trás o tempo atual. O código em "time.asp" olha como está:

```
<%
response.expires=-1
response.write(time)
%>
```

Nota: Expiram os sets da propriedade quanto tempo (nos minutos) uma página cached em um browser antes que expire. Se um usuário retornar à mesma página antes que expire, a versão cached está indicada. (**Response.Expires = -1)** indica que a página nunca cached.

AJAX sugere o exemplo

Nós vimos que AJAX pode ser usado para criar aplicações mais interativas.

AJAX sugere o exemplo

No exemplo de AJAX abaixo nós demonstraremos como um Web page pode se comunicar com um web server em linha enquanto um usuário incorpora dados em um formulário padrão do HTML.

Exemplo explicado - o formulário do HTML

O formulário tem o seguinte código do HTML:

```
<form>
First Name:
<input type="text" id="txt1"
onkeyup="showHint(this.value)">
</form>
<p>Suggestions: <span id="txtHint"></span></p>
```

Como você pode ver é justo um formulário simples do HTML com um campo da entrada chamado "txt1" .

Um atributo do evento para o campo da entrada define uma função a ser provocada pelo evento do **onkeyup**.

O parágrafo abaixo do formulário contem uma extensão chamada "txtHint". A extensão é usada como um placeholder para os dados recuperados do web server.

Quando os dados de entradas do usuário, uma função chamaram o "**showHint()"** é executado. A execução da função é provocada pelo evento do "**onkeyup**". Em outras

palavras: Cada vez que o usuário move seu dedo longe de uma chave de teclado dentro do campo da entrada, o **showHint** da função está chamado.

Exemplo explicado - a função do showHint()

A função do **showHint()** é uma função muito simples do Javascript colocada na seção do <head> da página HTML.

A função contem o seguinte código:

```
function showHint(str) {
  if (str.length==0) {
    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);
}
```

A função executa cada vez que um caráter é incorporado ao campo da entrada.

Se houver alguma entrada no campo do texto (str.length > 0) a função executa o seguinte:

- Define o URL (nome do arquivo) para emitir ao usuário
- Adiciona um parâmetro (q) ao URL com o índice do campo da entrada
- Adiciona um número aleatório para impedir que o usuário use um arquivo cached
- Cría um objeto de XMLHTTP, e diz o objeto para executar uma função chamada stateChanged quando uma mudança é provocada
- abre o objeto de XMLHTTP com o URL dado.
- Emite um pedido do HTTP ao usuário

Se o campo da entrada estiver vazio, a função cancela simplesmente o índice do placeholder do txtHint.

Exemplo explicado - a função de GetXmlHttpObject()

O exemplo acima chama uma função chamada **GetXmlHttpObject()**.

A finalidade da função é resolver o problema de criar objetos diferentes de **xmlHttp** para browsers diferentes.

A função é a listada abaixo:

```
function GetXmlHttpObject() {
  var xmlHttp=null;
  try {
    // Firefox, Opera 8.0+, Safari
    xmlHttp=new XMLHttpRequest();
  } catch (e) {
    // Internet Explorer
    try {
      xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
    } catch (e) {
      xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
  return xmlHttp;
}
```

Exemplo explicado - A função stateChanged()

A função **stateChanged()** contem o seguinte código:

```
function stateChanged() {
  if (xmlHttp.readyState==4) {
    document.getElementById("txtHint").innerHTML=xmlHttp.responseText;
  }
}
```

A função **stateChanged()** executa cada vez que o estado do objeto de **XMLHTTP** muda.

Quando o estado mudar a 4 (“completo”), o índice do placeholder do txtHint está enchido com o texto da resposta.

AJAX sugere o código fonte

Código fonte de AJAX para sugerir o exemplo

O código fonte abaixo pertence ao exemplo de AJAX na página precedente.

Você pode cópia e colá-lo, e tenta-o você mesmo.

O HTML da página de AJAX

Este é o HTML page. Contem um formulário simples do HTML e uma ligação a um Javascript.

```
<html>
<head>
<script src="clienthint.js"></script>
</head>
<body>
<form>
```

```
First Name:
<input type="text" id="txt1"
onkeyup="showHint(this.value)">
</form>
<p>Suggestions: <span id="txtHint"></span></p>
</body>
</html>
```

O código do Javascript é listado abaixo.

O AJAX JavaScript

Este é o código do Javascript, armazenado no arquivo “clienthint.js”:

```
var xmlHttp;

function showHint(str) {
  if (str.length==0) {
    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==4) {
    document.getElementById("txtHint").innerHTML=xmlHttp.responseText;
  }
}

function GetXmlHttpObject() {
  var xmlHttp=null;
  try {
    // Firefox, Opera 8.0+, Safari
    xmlHttp=new XMLHttpRequest();
  } catch (e) {
    // Internet Explorer
    try {
      xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
    } catch (e) {
      xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
  return xmlHttp;
}
```

A página do usuário de AJAX - ASP e PHP

Não há nenhuma coisa como um usuário de AJAX. As páginas de AJAX podem ser servidas por todo o usuário da Internet.

A página do usuário chamada pelo Javascript no exemplo do capítulo precedente é um arquivo simples do ASP chamada “gethint.asp”.

Abaixo nós listamos dois exemplos do código, de um escrito no ASP e de um da página do usuário em PHP.

Exemplo de AJAX ASP

O código na página de “gethint.asp” é escrito em VBScript para um usuário de informação do Internet (IIS). Ele verificações justas uma disposição dos nomes e dos retornos que corresponder nomeia ao cliente:

```
<%
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
%>
```

Exemplo de AJAX PHP

O código acima foi reescrito em PHP.

Nota: Para funcionar o exemplo inteiro em PHP, recordar mudar o valor da variável do URL em “clienthint.js” de “gethint.asp” a “gethint.php”.

Exemplo PHP

```
<?php
header("Cache-Control: no-cache, must-revalidate");
 // Date in the past
header("Expires: Mon, 26 Jul 1997 05:00:00 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";
//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;
?>
```

Exemplo da base de dados de AJAX

AJAX pode ser usado para uma comunicação interativa com uma base de dados.

Exemplo da base de dados de AJAX

No exemplo de AJAX abaixo nós demonstraremos como um Web page pode buscar a informação de uma base de dados usando a tecnologia de AJAX.

O exemplo de AJAX explicou

O exemplo contém um formulário simples do HTML e uma ligação a um Javascript:

```
<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> O cliente info será listado aqui.</b></div>
</p>
</body>
</html>
```

Como você pode ver é justo um formulário simples do HTML com uma gota encaixota para baixo “Customer”.

O parágrafo abaixo do formulário contem um “txtHint chamado div”. O div é usado como um placeholder para o info recuperado do web server.

Quando o usuário seleciona dados, uma função chamada “showCustomer ()” está executada. A execução da função é provocada pelo evento do “onchange”. Em outras palavras: Cada vez que a mudança que do usuário o valor na gota encaixota para baixo, o showCustomer da função é chamado.

O código do Javascript é alistado abaixo.

O AJAX JavaScript

Este é o código do Javascript armazenado no arquivo “selectcustomer.js”:

```
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==4)
{
document.getElementById("txtHint").innerHTML=xmlHttp.responseText;
}
}

function GetXmlHttpObject()
{
var xmlHttp=null;
try
  {
  // Firefox, Opera 8.0+, Safari
  xmlHttp=new XMLHttpRequest();
  }
catch (e)
  {
  // Internet Explorer
  try
    {
    xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
    }
  catch (e)
    {
    xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
return xmlHttp;
}
```

A página do usuário de AJAX

A página do usuário chamada pelo Javascript, é um arquivo simples do ASP chamada “getcustomer.asp”.

A página é escrita em VBScript para um usuário de informação do Internet (IIS). Poderia fàcilmente ser reescrita em PHP, ou em alguma outra língua do usuário.

O código funciona um SQL de encontro a uma base de dados e retorna o resultado como uma tabela do HTML:

```
<%
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>")
%>
```

Exemplo da base de dados de PHP, AJAX e MySQL

AJAX pode ser usado para uma comunicação interativa com uma base de dados.

Exemplo da base de dados de AJAX com PHP

No exemplo de AJAX abaixo nós demonstraremos como uma página web pode buscar a informação de uma base de dados de MySQL usando a tecnologia de AJAX.

Este exemplo consiste em quatro elementos:

- Base de dados MySQL
- Um formulário simples em html
- Um Javascript
- Uma página de PHP

A base de dados

A base de dados que nós nos estaremos usando em olhares deste exemplo como está:

id	FirstName	LastName	Age	Hometown	Job
1	Peter	Griffin	41	Quahog	Brewery
2	Lois	Griffin	40	Newport	Piano Teacher
3	Joseph	Swanson	39	Quahog	Police Officer
4	Glenn	Quagmire	41	Quahog	Pilot

O formulário do HTML

O exemplo acima contem um formulário simples do HTML e uma ligação a um Javascript:

```
<html>
<head>
<script src="selectuser.js"></script>
</head>
<body>
<form>
Select a User:
<select name="users" onchange="showUser(this.value)">
<option value="1">Peter Griffin</option>
<option value="2">Lois Griffin</option>
```

```
<option value="3">Glenn Quagmire</option>
<option value="4">Joseph Swanson</option>
</select>
</form>
<p>
<div id="txtHint"><b>User info will be listed here.</b></div>
</p>
</body>
</html>
```

Exemplo explicado - o formulário do HTML

Como você pode ver é justo um formulário simples do HTML com uma gota encaixota para baixo “usuários chamados” com nomes e a “identificação” da base de dados como valores da opção.

O parágrafo abaixo do formulário contem um “txtHint chamado div”. O div é usado como um placeholder para o info recuperado do web server.

Quando o usuário seleciona dados, uma função chamada “**showUser()**” está executada. A execução da função é provocada pelo evento do “**onchange**”.

Em outras palavras: Cada vez que o usuário muda o valor do campo select, o **showUser()** da função está chamado.

O JavaScript

Este é o código do Javascript armazenado no arquivo “selectuser.js”:

```
var xmlHttp
function showUser(str) {
  xmlHttp=GetXmlHttpObject()
  if (xmlHttp==null) {
    alert ("Browser does not support HTTP Request")
    return
  }
  var url="getuser.php"
  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==4 || xmlHttp.readyState=="complete") {
    document.getElementById("txtHint").innerHTML=xmlHttp.responseText
  }
}
function GetXmlHttpObject() {
  var xmlHttp=null;
  try {
    // Firefox, Opera 8.0+, Safari
    xmlHttp=new XMLHttpRequest();
  } catch (e) {
    //Internet Explorer
```

```
    try {
      xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
    } catch (e) {
      xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
  return xmlHttp;
}
```

O exemplo explicou

stateChanged() e as funções de **GetXmlHttpObject**() são as mesmas que no AJAX de PHP sugerir o capítulo.

A função do showUser()

Se um artigo for selecionado a função executar o seguinte:

1. Convida a função de **GetXmlHttpObject** para criar um do objeto de **xmlHttp**
2. Define o URL (nome do arquivo) para emitir ao do usuário
3. Adiciona um parâmetro (q) ao URL com o índice da caixa dropdown
4. Adiciona um número aleatório para impedir que o usuário use um arquivo cached
5. A chamada **stateChanged** quando uma mudança é provocada
6. Abre o objeto de **xmlHttp** com o URL dado.
7. Emite um pedido do HTTP ao usuário

A página PHP

A página do usuário chamou-se pelo Javascript, é um arquivo simples de PHP chamada "getuser.php".

A página é escrita em PHP e usa um banco de dados de MySQL.

O código funciona com uma pergunta ao MySQL de encontro a uma base de dados e retorna o resultado como uma tabela do HTML:

```
<?php
$q=$_GET["q"];

$con = mysql_connect('localhost', 'peter', 'abc123');
if (!$con)
 {
 die('Could not connect: ' . mysql_error());
 }

mysql_select_db("ajax_demo", $con);

$sql="SELECT * FROM user WHERE id = '".$q."'";

$result = mysql_query($sql);

echo "<table border='1'>
<tr>
<th>Firstname</th>
```

```
<th>Lastname</th>
<th>Age</th>
<th>Hometown</th>
<th>Job</th>
</tr>";

while($row = mysql_fetch_array($result))
 {
 echo "<tr>";
 echo "<td>" . $row['FirstName'] . "</td>";
 echo "<td>" . $row['LastName'] . "</td>";
 echo "<td>" . $row['Age'] . "</td>";
 echo "<td>" . $row['Hometown'] . "</td>";
 echo "<td>" . $row['Job'] . "</td>";
 echo "</tr>";
 }
echo "</table>";

mysql_close($con);
?>
```

Exemplo explicado

Quando a pergunta é emitida do Javascript à página de PHP o seguinte acontece:

1. PHP abre uma conexão a um usuário de MySQL
2. O “user” com o nome especificado é encontrado
3. Uma tabela é criada e os dados são introduzidos e emitidos ao placeholder do “txtHint”

Exemplo de AJAX XML

AJAX pode ser usado para uma comunicação interativa com um arquivo de XML.

Exemplo de AJAX XML

No exemplo de AJAX abaixo nós demonstraremos como um Web page pode buscar a informação de um arquivo de XML usando a tecnologia de AJAX.

AJAX Example Explained

O exemplo um formulário simples do HTML e uma ligação a um Javascript:

```
<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>
```

Como você pode ver é justo um formulário simples do HTML com uma gota simples encaixota para baixo “CD chamados”.

O parágrafo abaixo do formulário contem um “txtHint chamado div”. O div é usado como um placeholder para o info recuperado do web server.

Quando o usuário seleciona dados, uma função chamada “showCD” está executada. A execução da função é provocada pelo evento do “onchange”. Em outras palavras: Cada vez que a mudança que do usuário o valor na gota encaixota para baixo, o showCD da função é chamado.

O código do Javascript é alistado abaixo.

O AJAX JavaScript

Este é o código do Javascript armazenado no arquivo “selectcd.js”:

```
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==4)
{
document.getElementById("txtHint").innerHTML=xmlHttp.responseText;
}
}
function GetXmlHttpObject()
{
var xmlHttp=null;
try
  {
```

```
  // Firefox, Opera 8.0+, Safari
  xmlHttp=new XMLHttpRequest();
  }
catch (e)
  {
  // Internet Explorer
  try
    {
    xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
    }
  catch (e)
    {
    xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
return xmlHttp;
}
```

A página do usuário de AJAX

A página do usuário chamada pelo Javascript, é um arquivo simples do ASP chamada "getcd.asp".

A página é escrita em VBScript para um usuário de informação do Internet (IIS). Poderia fàcilmente ser reescrita em PHP, ou em alguma outra língua do usuário.

O código funciona uma pergunta de encontro a um arquivo de XML e retorna o resultado como o HTML:

```
<%
response.expires=-1
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
%>
```

Exemplo de PHP e de AJAX XML

AJAX pode ser usado para uma comunicação interativa com um arquivo de XML.

Exemplo de AJAX XML

No exemplo de AJAX abaixo nós demonstraremos como uma página web pode buscar a informação de um arquivo de XML usando a tecnologia de AJAX.

Este exemplo consiste em quatro páginas:

- Um formulário simples do HTML
- Uma arquivo XML
- Um JavaScript
- Uma página em PHP

O formulário do HTML

O exemplo acima contem um formulário simples do HTML e uma ligação a um Javascript:

```
<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="Bee Gees">Bee Gees</option>
<option value="Cat Stevens">Cat Stevens</option>
</select>
</form>
<p>
<div id="txtHint"><b>CD info will be listed here.</b></div>
</p>
</body>
</html>
```

Exemplo explicado

Como você pode ver é justo um formulário simples do HTML com um select simples “chamados CD”.

O parágrafo abaixo do formulário contem um “txtHint chamado div”. O div é usado como um placeholder para o info recuperado do web server.

Quando o usuário seleciona dados, uma função chamada “showCD” está executada. A execução da função é provocada pelo evento do “onchange”.

Em outras palavras: Cada vez que o usuário muda o valor do campo select, o showCD da função está chamado.

O arquivo XML

Aqui está o código do arquivo XML de nome “cd_catalog.xml”.

```
?xml version="1.0" encoding="ISO-8859-1" ?>
<!--
dited with XML Spy v2007 (http://www.altova.com)
-->
ATALOG>
D>
ITLE>Empire Burlesque</TITLE>
RTIST>Bob Dylan</ARTIST>
OUNTRY>USA</COUNTRY>
OMPANY>Columbia</COMPANY>
RICE>10.90</PRICE>
EAR>1985</YEAR>
</CD>
D>
ITLE>Hide your heart</TITLE>
RTIST>Bonnie Tyler</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>CBS Records</COMPANY>
RICE>9.90</PRICE>
EAR>1988</YEAR>
</CD>
D>
ITLE>Greatest Hits</TITLE>
RTIST>Dolly Parton</ARTIST>
OUNTRY>USA</COUNTRY>
OMPANY>RCA</COMPANY>
RICE>9.90</PRICE>
EAR>1982</YEAR>
</CD>
D>
ITLE>Still got the blues</TITLE>
RTIST>Gary Moore</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>Virgin records</COMPANY>
RICE>10.20</PRICE>
EAR>1990</YEAR>
</CD>
D>
ITLE>Eros</TITLE>
RTIST>Eros Ramazzotti</ARTIST>
OUNTRY>EU</COUNTRY>
OMPANY>BMG</COMPANY>
RICE>9.90</PRICE>
EAR>1997</YEAR>
</CD>
D>
ITLE>One night only</TITLE>
RTIST>Bee Gees</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>Polydor</COMPANY>
RICE>10.90</PRICE>
EAR>1998</YEAR>
</CD>
D>
ITLE>Sylvias Mother</TITLE>
RTIST>Dr.Hook</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>CBS</COMPANY>
RICE>8.10</PRICE>
EAR>1973</YEAR>
</CD>
D>
ITLE>Maggie May</TITLE>
RTIST>Rod Stewart</ARTIST>
```

```
OUNTRY>UK</COUNTRY>
OMPANY>Pickwick</COMPANY>
RICE>8.50</PRICE>
EAR>1990</YEAR>
</CD>
D>
ITLE>Romanza</TITLE>
RTIST>Andrea Bocelli</ARTIST>
OUNTRY>EU</COUNTRY>
OMPANY>Polydor</COMPANY>
RICE>10.80</PRICE>
EAR>1996</YEAR>
</CD>
D>
ITLE>When a man loves a woman</TITLE>
RTIST>Percy Sledge</ARTIST>
OUNTRY>USA</COUNTRY>
OMPANY>Atlantic</COMPANY>
RICE>8.70</PRICE>
EAR>1987</YEAR>
</CD>
D>
ITLE>Black angel</TITLE>
RTIST>Savage Rose</ARTIST>
OUNTRY>EU</COUNTRY>
OMPANY>Mega</COMPANY>
RICE>10.90</PRICE>
EAR>1995</YEAR>
</CD>
D>
ITLE>1999 Grammy Nominees</TITLE>
RTIST>Many</ARTIST>
OUNTRY>USA</COUNTRY>
OMPANY>Grammy</COMPANY>
RICE>10.20</PRICE>
EAR>1999</YEAR>
</CD>
D>
ITLE>For the good times</TITLE>
RTIST>Kenny Rogers</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>Mucik Master</COMPANY>
RICE>8.70</PRICE>
EAR>1995</YEAR>
</CD>
D>
ITLE>Big Willie style</TITLE>
RTIST>Will Smith</ARTIST>
OUNTRY>USA</COUNTRY>
OMPANY>Columbia</COMPANY>
RICE>9.90</PRICE>
EAR>1997</YEAR>
</CD>
D>
ITLE>Tupelo Honey</TITLE>
RTIST>Van Morrison</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>Polydor</COMPANY>
RICE>8.20</PRICE>
EAR>1971</YEAR>
</CD>
D>
ITLE>Soulsville</TITLE>
RTIST>Jorn Hoel</ARTIST>
OUNTRY>Norway</COUNTRY>
OMPANY>WEA</COMPANY>
```

```
RICE>7.90</PRICE>
EAR>1996</YEAR>
</CD>
D>
ITLE>The very best of</TITLE>
RTIST>Cat Stevens</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>Island</COMPANY>
RICE>8.90</PRICE>
EAR>1990</YEAR>
</CD>
D>
ITLE>Stop</TITLE>
RTIST>Sam Brown</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>A and M</COMPANY>
RICE>8.90</PRICE>
EAR>1988</YEAR>
</CD>
D>
ITLE>Bridge of Spies</TITLE>
RTIST>T'Pau</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>Siren</COMPANY>
RICE>7.90</PRICE>
EAR>1987</YEAR>
</CD>
D>
ITLE>Private Dancer</TITLE>
RTIST>Tina Turner</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>Capitol</COMPANY>
RICE>8.90</PRICE>
EAR>1983</YEAR>
</CD>
D>
ITLE>Midt om natten</TITLE>
RTIST>Kim Larsen</ARTIST>
OUNTRY>EU</COUNTRY>
OMPANY>Medley</COMPANY>
RICE>7.80</PRICE>
EAR>1983</YEAR>
</CD>
D>
ITLE>Pavarotti Gala Concert</TITLE>
RTIST>Luciano Pavarotti</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>DECCA</COMPANY>
RICE>9.90</PRICE>
EAR>1991</YEAR>
</CD>
D>
ITLE>The dock of the bay</TITLE>
RTIST>Otis Redding</ARTIST>
OUNTRY>USA</COUNTRY>
OMPANY>Atlantic</COMPANY>
RICE>7.90</PRICE>
EAR>1987</YEAR>
</CD>
D>
ITLE>Picture book</TITLE>
RTIST>Simply Red</ARTIST>
OUNTRY>EU</COUNTRY>
OMPANY>Elektra</COMPANY>
RICE>7.20</PRICE>
EAR>1985</YEAR>
```

```
</CD>
D>
ITLE>Red</TITLE>
RTIST>The Communards</ARTIST>
OUNTRY>UK</COUNTRY>
OMPANY>London</COMPANY>
RICE>7.80</PRICE>
EAR>1987</YEAR>
</CD>
D>
ITLE>Unchain my heart</TITLE>
RTIST>Joe Cocker</ARTIST>
OUNTRY>USA</COUNTRY>
OMPANY>EMI</COMPANY>
RICE>8.20</PRICE>
EAR>1987</YEAR>
</CD>
</CATALOG>
```

O JavaScript

Aqui fica o código JavaScript no arquivo "selectcd.js":

```
var xmlHttp

function showCD(str)
{
xmlHttp=GetXmlHttpObject()
if (xmlHttp==null)
 {
 alert ("Browser does not support HTTP Request")
 return
 }
var url="getcd.php"
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==4 || xmlHttp.readyState=="complete")
 {
 document.getElementById("txtHint").innerHTML=xmlHttp.responseText
 }
}
function GetXmlHttpObject()
{
var xmlHttp=null;
try
 {
 // Firefox, Opera 8.0+, Safari
 xmlHttp=new XMLHttpRequest();
 }
catch (e)
 {
 // Internet Explorer
 try
```

```
  {
  xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
  }
 catch (e)
  {
  xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
 }
return xmlHttp;
}
```

Exemplo explicado

stateChanged() e as funções de **GetXmlHttpObject** são as mesmas que no último capítulo.

A função showCD()

Se um select for selecionado a função executar o seguinte:

1. Convida a função de **GetXmlHttpObject** para criar um objeto de **xmlHttp**
2. Define o URL (nome do arquivo) para emitir ao usuário
3. Adiciona um parâmetro (q) ao URL com o índice do campo da entrada
4. Adiciona um número aleatório para impedir que o usuário use um arquivo cached
5. A chamada **stateChanged** quando uma mudança é provocada
6. Abre o objeto de **XMLHTTP** com o URL dado.
7. Emite um pedido do HTTP ao usuário

A página PHP

O usuário paginou chamado pelo Javascript, é um arquivo simples de PHP chamada “getcd.php”.

A página é escrita em PHP usando o XML DOM carregar o original “cd_catalog.xml” de XML.

O código funciona uma pergunta de encontro ao arquivo XML e retorna o resultado como o HTML:

```
<?php
$q=$_GET["q"];
$xmlDoc = new DOMDocument();
$xmlDoc->load("cd_catalog.xml");
$x=$xmlDoc->getElementsByTagName('ARTIST');
for ($i=0; $i<=$x->length-1; $i++)
{
//Process only element nodes
if ($x->item($i)->nodeType==1)
  {
  if ($x->item($i)->childNodes->item(0)->nodeValue == $q)
    {
    $y=($x->item($i)->parentNode);
```

```
    }
  }
}
$cd=($y->childNodes);
for ($i=0;$i<$cd->length;$i++)
{
//Process only element nodes
if ($cd->item($i)->nodeType==1)
  {
  echo($cd->item($i)->nodeName);
  echo(": ");
  echo($cd->item($i)->childNodes->item(0)->nodeValue);
  echo("<br />");
  }
}
?>
```

Exemplo explicado

Quando a pergunta é emitida do Javascript à página de PHP o seguinte acontece:

1. PHP cría um objeto de XML DOM do arquivo de “cd_catalog.xml”.
2. Todos os elementos do “artista” (nodetypes = 1) são dados laços completamente para encontrar um nome combinar esse emitido do Javascript.
3. O CD que contem o artista correto é encontrado.
4. A informação do album é emitida ao placeholder do “txtHint”.

Exemplo de AJAX ResponseXML

Quando o responseText retornar a resposta do HTTP como uma string, o responseXML retorna a resposta como XML.

A propriedade de ResponseXML retorna um objeto do original de XML, que possa ser examinado e analisado gramaticalmente usando métodos e propriedades da árvore do nó de W3C DOM.

Exemplo de AJAX ResponseXML

No seguinte exemplo de AJAX nós demonstraremos como um Web page pode buscar a informação de uma base de dados usando a tecnologia de AJAX. Os dados selecionados da base de dados esta vez serão convertidos a um original de XML, e então nós usaremos o DOM extrair os valores a ser indicados.

Exemplo de AJAX explicado

O exemplo acima contem um formulário do HTML, diversos elementos do para prender os dados retornados, e um link a um Javascript:

```
<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>
```

O exemplo acima contem um formulário do HTML com uma gota encaixota para baixo “clientes chamados”.

Quando o usuário seleciona um cliente na caixa dropdown, uma função chamada “showCustomer()” está executada. A execução da função é provocada pelo evento do “onchange”. Em outras palavras: Cada vez que a mudança que do usuário o valor na gota encaixota para baixo, o showCustomer() da função é chamado.

O código do Javascript é alistado abaixo.

O AJAX JavaScript

Este é o código do Javascript armazenado no arquivo “selectcustomer_xml.js”:

```
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==4)
{
var xmlDoc=xmlHttp.responseXML.documentElement;
document.getElementById("companyname").innerHTML=
xmlDoc.getElementsByTagName("compname")[0].childNodes[0].nodeValue;
document.getElementById("contactname").innerHTML=
xmlDoc.getElementsByTagName("contname")[0].childNodes[0].nodeValue;
```

```
document.getElementById("address").innerHTML=
xmlDoc.getElementsByTagName("address")[0].childNodes[0].nodeValue;
document.getElementById("city").innerHTML=
xmlDoc.getElementsByTagName("city")[0].childNodes[0].nodeValue;
document.getElementById("country").innerHTML=
xmlDoc.getElementsByTagName("country")[0].childNodes[0].nodeValue;
}
}

function GetXmlHttpObject()
{
var xmlHttp=null;
try
  {
  // Firefox, Opera 8.0+, Safari
  xmlHttp=new XMLHttpRequest();
  }
catch (e)
  {
  // Internet Explorer
  try
    {
    xmlHttp=new ActiveXObject("Msxml2.XMLHTTP");
    }
  catch (e)
    {
    xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  }
return xmlHttp;
}
```

O showCustomer() e as funções de GetXmlHttpObject() acima estão o mesmo que em capítulos precedentes. stateChanged() a função é usado também mais cedo neste tutorial, entretanto; esta vez nós retornamos o resultado como um original de XML (com responseXML) e usos o DOM extrair os valores que nós queremos ser indicados.

A página do usuário de AJAX

A página do usuário chamada pelo Javascript, é um arquivo simples do ASP chamada “getcustomer_xml.asp”.

A página é escrita em VBScript para um usuário de informação do Internet (IIS). Poderia fàcilmente ser reescrita em PHP, ou em alguma outra língua do usuário.

O código funciona uma pergunta do SQL de encontro a uma base de dados e retorna o resultado como um original de XML:

```
<%
response.expires=-1
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.4.0"
conn.Open(Server.Mappath("/db/northwind.mdb"))
set rs=Server.CreateObject("ADODB.recordset")
rs.Open sql, conn
if err <> 0 then
response.write(err.description)
set rs=nothing
set conn=nothing
else
response.write("<?xml version='1.0' encoding='ISO-8859-1'?>")
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 0
%>
```

Observar a segunda linha no código do ASP acima: response.contenttype= " texto/xml ". A propriedade de ContentType ajusta o tipo satisfeito do HTTP para o objeto da resposta. O valor de defeito para esta propriedade é o “texto/HTML”. Esta vez nós queremos o tipo satisfeito ser XML. Então nós selecionamos dados da base de dados, e configurações um original de XML com os dados.

Exemplo do responseXML de PHP e de AJAX

AJAX pode ser usado retornar a informação da base de dados como XML.

Base de dados de AJAX como o exemplo de XML

No exemplo de AJAX abaixo nós demonstraremos como um página web pode buscar a informação de uma base de dados de MySQL, a converter a um original de XML, e a usar à informação de exposição em diversos lugares diferentes.

Este exemplo meu parece muito como “o exemplo da base de dados de AJAX PHP” no último capítulo, porém há uma diferença grande: neste exemplo nós começamos os dados da página de PHP enquanto XML que usam o responseXML funcionam.

Receber a resposta como um original de XML permite que nós atualizemos esta página em diversos lugares, em vez apenas de receber um PHP output e de indicá-lo.

Neste exemplo nós atualizaremos diversos elementos do com a informação que nós recebemos da base de dados.

Este exemplo consiste em quatro elementos:

- Uma base de dados de MySQL
- Um Formulário simples em HTML
- Um JavaScript

- Uma página em PHP

A base de dados

A base de dados que nós nos estaremos usando em olhares deste exemplo como este:

id	FirstName	LastName	Age	Hometown	Job
1	Peter	Griffin	41	Quahog	Brewery
2	Lois	Griffin	40	Newport	Piano Teacher
3	Joseph	Swanson	39	Quahog	Police Officer
4	Glenn	Quagmire	41	Quahog	Pilot

O formulário do HTML

O exemplo acima contem um formulário simples do HTML e uma ligação a um Javascript:

```
<html>
<head>
<script src="responsexml.js"></script>
</head>
<body>
<form>
Select a User:
<select name="users" onchange="showUser(this.value)">
<option value="1">Peter Griffin</option>
<option value="2">Lois Griffin</option>
<option value="3">Glenn Quagmire</option>
<option value="4">Joseph Swanson</option>
</select>
</form>
<h2><span id="firstname"></span>
 <span id="lastname"></span></h2>
<span id="job"></span>
<div style="text-align: right">
<span id="age_text"></span>
<span id="age"></span>
<span id="hometown_text"></span>
<span id="hometown"></span>
</div>
</body>
</html>
```

Exemplo explicado - o formulário do HTML

- O formulário do HTML é um select “users” com nomes e a “identificação” da base de dados como valores da opção.
- Abaixo do formulário há diversos elementos diferentes do a que são usados porque placeholders para os valores diferentes nós vontade retrive.
- Quando o usuário seleciona um objeto, uma função chamada “**showUser()**” está executada. A execução da função é provocada pelo evento do “**onchange**”.

Em outras palavras: Cada vez que o usuário muda o valor no campo select, o **showUser()** da função é chamado e outputs o resultado nos elementos especificados do .

O JavaScript

Este é o código do Javascript armazenado no arquivo “responsexml.js”:

```
var xmlHttp
function showUser(str) {
  xmlHttp=GetXmlHttpObject()
  if (xmlHttp==null){
    alert ("Browser does not support HTTP Request")
    return
  }
  var url="responsexml.php"
  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==4 || xmlHttp.readyState=="complete") {
    xmlDoc=xmlHttp.responseXML;
    document.getElementById("firstname").innerHTML=
    xmlDoc.getElementsByTagName("firstname")[0].childNodes[0].nodeValue;
    document.getElementById("lastname").innerHTML=
    xmlDoc.getElementsByTagName("lastname")[0].childNodes[0].nodeValue;
    document.getElementById("job").innerHTML=
    xmlDoc.getElementsByTagName("job")[0].childNodes[0].nodeValue;
    document.getElementById("age_text").innerHTML="Age: ";
    document.getElementById("age").innerHTML=
    xmlDoc.getElementsByTagName("age")[0].childNodes[0].nodeValue;
    document.getElementById("hometown_text").innerHTML="<br/>From: ";
    document.getElementById("hometown").innerHTML=
    xmlDoc.getElementsByTagName("hometown")[0].childNodes[0].nodeValue;
  }
}
function GetXmlHttpObject() {
  var objXMLHttp=null
  if (window.XMLHttpRequest) {
    objXMLHttp=new XMLHttpRequest()
  } else if (window.ActiveXObject) {
    objXMLHttp=new ActiveXObject("Microsoft.XMLHTTP")
  }
  return objXMLHttp
}
```

O exemplo explicou

O **showUser()** e as funções de **GetXmlHttpObject** são o mesmo que no capítulo da base de dados de AJAX de PHP.

A função stateChanged()

Se um artigo no select selecionado a função executar o seguinte:

1. Define a variável do “xmlDoc” como um original do xml usando a função do responseXML
2. Recupera dados dos originais do xml e coloca-os nos elementos corretos do

A página de PHP

A página do usuário chamou-se pelo Javascript, é um arquivo simples de PHP chamada “responsexml.php”.

A página é escrita em PHP e usa um databse de MySQL.

O código funciona uma pergunta do SQL de encontro a uma base de dados e retorna o resultado como um original de XML:

```
<?php
header('Content-Type: text/xml');
header("Cache-Control: no-cache, must-revalidate");
//A date in the past
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT");
$q=$_GET["q"];
$con = mysql_connect('localhost', 'peter', 'abc123');
if (!$con)
 {
 die('Could not connect: ' . mysql_error());
 }
mysql_select_db("ajax_demo", $con);
$sql="SELECT * FROM user WHERE id = ".$q."";
$result = mysql_query($sql);
echo '<?xml version="1.0" encoding="ISO-8859-1"?>
<person>';
while($row = mysql_fetch_array($result))
 {
 echo "<firstname>" . $row['FirstName'] . "</firstname>";
 echo "<lastname>" . $row['LastName'] . "</lastname>";
 echo "<age>" . $row['Age'] . "</age>";
 echo "<hometown>" . $row['Hometown'] . "</hometown>";
 echo "<job>" . $row['Job'] . "</job>";
 }
echo "</person>";
mysql_close($con);
?>
```

O exemplo explicou

Quando a pergunta é emitida do Javascript à página de PHP o seguinte acontece:

1. O índice-tipo do original de PHP é ajustado para ser “texto/xml”
2. O original de PHP é ajustado ao “nenhum-esconderijo” para impedir caching
3. A variável de $q é ajustada para ser os dados emitidos do HTML page
4. PHP abre uma conexão a um usuário de MySQL
5. O “usuário” com a identificação especificada é encontrado
6. Os dados são outputted como um original do xml

AJAX AppML

AppML é uma iniciativa aberta da fonte de W3Schools.

AppML usa a tecnologia de AJAX.

Que é AppML?

- AppML está para Application Markup Language
- AppML usa XML para descrever aplicações da Internet
- As aplicações de AppML são self-describing
- AppML é uma língua declarativa
- AppML é uma plataforma independente
- AppML usa a tecnologia de AJAX
- AppML é uma iniciativa open source(fonte aberta) da W3Schools

AppML é uma língua declarativa

AppML não é uma língua de programação. É uma língua declarativa, usada descrever aplicações.

Com o AppML você pode criar aplicações do Internet sem programar.

As aplicações tradicionais são escritas em uma língua de programação e compiladas, com estruturas de dados e funções predefinidas. AppML permite que o programador redefina dados e funções quando a aplicação funcionar.

Desde que as aplicações de AppML são escritas em XML, as aplicações de AppML são self-describing.

AppML é Independent do Browser

Desde que AppML usa somente padrões da Internet como HTML (XHTML), CSS, XML, e Javascript, AppML funcionará em todos os browsers.

AppML usa a tecnologia de AJAX

AppML usa a tecnologia de AJAX. Uma comunicação da Internet entre o web client e o web server é feita com os pedidos do HTTP.

Historia

Em 1999, a equipe de funcionários em W3Schools começou a desenvolver AppML.

Em setembro 2000, um projeto do desenvolvimento para um cliente norueguês grande foi começado. O objetivo do projeto era converter um sistema de informação enorme de uma aplicação desktop de Windows a uma aplicação moderna da Internet usando somente AppML.

O sistema AppML-baseado novo foi lançado em 2001, diversos meses antes da programação, era uma das aplicações disponíveis de AJAX do primeiro comercial.

O projeto era um sucesso enorme, com o tempo de desenvolvimento reduzido por 75% comparado ao desenvolvimento ordinário da web applications.

Desde então, as centenas de aplicações novas têm sido adicionadas, e as tampas de AppML agora sobre 1000 aplicações running.

Em dezembro 2006, W3Schools decidiu oferecer AppML ao público, como um produto de fonte aberta , livre da carga.