O que é o AJAX e como ele funciona

Publicado por: Everton da Rosa 21 de setembro de 2009 em Artigos 1 Comentário

O que é AJAX?

AJAX é carregar e renderizar uma página, utilizando recursos de scripts rodando pelo lado cliente, buscando e carregando dados em background sem a necessidade de reload da página.
AJAX é acrônimo para: Asynchronous JavaScript And XML e foi gerado por Jesse James Garret, em um artigo no site

http://www.adaptivepath.com/publications/essays/archives/000385.php, da sua empresa Adaptive Path, em fevereiro de 2005.
Ajax não é uma tecnologia, mas sim um conjunto de tecnologias. O conceito de AJAX se resume em conhecer bem JavaScript, trabalhando com DOM (Document Object Model), CSS (Cascading Style Sheets) e XML.

Como o AJAX trabalha

Enquanto em uma aplicação Web clássica o navegador tem que ir buscar as informações no servidor e retornar para o cliente, no Ajax ocorre de forma diferente. No carregamento da página, toda a lógica de processamento de dados é passado ao cliente. Quando o usuário faz uma requisição, quem busca e trás essas informações é o JavaScript, de forma assíncrona, não causando assim o chamado “reload” na tela. O tratamento dos dados, seu formato e exibição fica toda por conta do script que foi carregado inicialmente quando se acessou a página. O processo inicial de carregamento é mais lento que de uma aplicação comum, pois muitas informações são pré-carregadas. Mas depois, somente os dados são carregados, tornando assim o site mais rápido.

Criando uma página com Ajax

Com o uso do objeto *XMLHttpRequest*, que faz parte do padrão ECMA e está presente em todas as boas versões do Javascript. Os browsers que suportam esse padrão são:

- Opera 8
- Mozilla e Firefox
- Konqueror
- Safari
- Além disso o Internet Explorer, desde a versão 5, suporta o Microsoft XMLHTTP, um substituto para o *XMLHttpRequest*.

Há duas maneiras de se fazer uma requisição com um objeto *XMLHttpRequest*, uma é síncrona, outra assíncrona. No modo síncrono, quando você manda o objeto fazer uma requisição, o seu script é interrompido esperando pelo retorno. No modo assíncrono a requisição é feita em segundo plano e seu script continua a ser executado. Em modo síncrono, você tem o problema de ter seu navegador congelado enquanto seu script é executado. E isso é ruim, pois podem ser que seja rápida a requisição e pode ser que não, ai você pergunta, será que está funcionando ou travou? O negócio é evitar esse método.

Exemplo

ajax.php

```php
<?php
try{
 //conecta ao banco de dados
 $conexao=mysql_connect("localhost","edson","integrator");
 //acessa o banco de dados desejado
 $banco=mysql_select_db("livraria");
 $rs = mysql_query("SELECT * FROM livros");
 if(!$rs)
 throw new Exception('Problemas: '.mysql_error().'<br />');
}
catch(Exception $e){
```

```
 //caso haja uma exceção a mensagem é capturada e atribuida a $msg
 $msg = $e->getMessage();
}
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Trabalhando com Ajax</title>
<script language="JavaScript">

function Dados(isbn) {
 //verifica se o browser tem suporte a ajax
 try {
  ajax = new ActiveXObject("Microsoft.XMLHTTP");
 }
 catch(e) {
  try {
   ajax = new ActiveXObject("Msxml2.XMLHTTP");
  }
  catch(ex) {
   try {
    ajax = new XMLHttpRequest();
   }
   catch(exc) {
    alert("Esse browser não tem recursos para uso do Ajax");
    ajax = null;
   }
  }
 }
 //se tiver suporte ajax
 if(ajax) {
  ajax.open("GET", "livro.php?isbn="+isbn, true);
  ajax.onreadystatechange = function() {
  //enquanto estiver processando...emite a msg de carregando
  if(ajax.readyState == 1) {
   mensagem( "Carregando...!" );
  }
  //após ser processado - chama função processXML que vai varrer os dados
 if(ajax.readyState == 4 ) {
  if(ajax.responseXML) {
   processXML(ajax.responseXML);
  }
  else {
   //caso não seja um arquivo XML emite a mensagem abaixo
   mensagem( "Erro ao carregar" );
  }
 }
}
 ajax.send(null);
}
}//end function Dados

function processXML(obj){
  //pega a tag livro do XML
  var dataArray = obj.getElementsByTagName("livro");
  //total de elementos contidos na tag livro
  if(dataArray.length > 0) {
   //percorre o arquivo XML paara extrair os dados
   for(var i = 0 ; i < dataArray.length ; i++) {
    var item = dataArray[i];
    //contéudo dos campos no arquivo XML
    var isbn = item.getElementsByTagName("isbn")[0].firstChild.nodeValue;
    var titulo = item.getElementsByTagName("titulo")[0].firstChild.nodeValue;
    var edicao = item.getElementsByTagName("edicao")[0].firstChild.nodeValue;
    var publicacao = item.getElementsByTagName("publicacao")[0].firstChild.nodeValue;
   }
   mensagem( "Dados carregados" );
   document.getElementById('isbn').innerHTML=isbn;
   document.getElementById('titulo').innerHTML=titulo;
   document.getElementById('edicao').innerHTML=edicao;
   document.getElementById('publicacao').innerHTML=publicacao;
  }
}//end function processXML

function mensagem(msg){
  document.getElementById('mensagem').innerHTML=msg;
}//end function mensagem

</script>
</head>
<body>
<?php
//verifica se existe a variável $msg
if(isset($msg))
 echo $msg;
?>
<table width="253" border="1" cellspacing="0" cellpadding="0">
<tr>
<th width="137" align="left">ISBN</th>
<th width="110" align="left">Exibir dados </th>
</tr>
```

```
<?php
 //varre todos os dados da tabela
 while($row=mysql_fetch_array($rs)){
?>
<tr>
<td><?php echo $row['isbn']?></td>
<td align="center">
<a href="#" onclick="Dados('<?php echo urlencode($row['isbn'])?>')">
Clique aqui </a> </td>
</tr>
<?php }//end if?>
</table>
<p>
<div id="mensagem"></div>
<table width="295" height="92" border="0" cellpadding="2" cellspacing="0">
<tr>
<td width="75">ISBN:</td>
<td width="212"><span id="isbn"></span></td>
</tr>
<tr>
<td>Título:</td>
<td><span id="titulo"></span></td>
</tr>
<tr>
<td>Edição N.º</td>
<td><span id="edicao"></span></td>
</tr>
<tr>
<td>Publicação:</td>
<td><span id="publicacao"></span></td>
</tr>
</table>
</p>
</body>
</html>
```

livro.php

```

<?php
 //conexao ao mysql
 $conexao=mysql_connect("localhost","edson","integrator");
 //acessa o banco de dados desejado
 $banco=mysql_select_db("livraria");
//recebendo o parâmetro
 $isbn = $_GET["isbn"];
 //executa a query
 $rs = mysql_query("SELECT * FROM livros WHERE isbn='$isbn'");
 //conta a quantidade de linhas encontradas
 $row = mysql_num_rows($rs);
 //se existem dados
 if($row>0) {
  //gera o xml
  $xml = "<?xml version="1.0" encoding="ISO-8859-1"?>n";
  $xml .= "<livros>n";
  //percorre os dados encontrados
  while($l=mysql_fetch_array($rs)){
    $xml .= "<livro>n";
    $xml .= "<isbn>".$l['isbn']."</isbn>n";
    $xml .= "<titulo>".$l['titulo']."</titulo>n";
    $xml .= "<edicao>".$l['edicao_num']."</edicao>n";
    $xml .= "<publicacao>".$l['ano_publicacao']."</publicacao>n";
    $xml .= "</livro>n";
  }//end while
  $xml.= "</livros>n";
  //saída para o navegador
  header("Content-type: application/xml; charset=iso-8859-1");
 }//end if
 //echo do resultado
 echo $xml;
?>
```

Entendendo o AJAX

Para ter a forma com que o objeto *XMLHttpRequest* vai trabalhar, você tem que alterar o terceiro parâmetro do método open.
Com esse parâmetro em true, no terceiro parâmetro do método open, coloca o objeto em modo assíncrono.
O método open do objeto *XMLHttpRequest* permite abrir um documento, passar argumentos para ele e capturar uma resposta.
Com apenas dois métodos possíveis de se utilizar para acessar um documento: GET e POST, o método usado no exemplo é GET. No entanto se a quantidade de informações a ser passada for muito grande você deverá alterar para o método POST.
O método send ativa a conexão e faz a requisição de informações ao documento aberto pelo método open. Este método possui somente um parâmetro que serve para enviar dados extras ao documento que está sendo acessado.
O browser Internet Explorer não o obriga a passar nenhum parâmetro, mas outros navegadores como o Mozilla, exige algum dado, neste caso, a solução foi enviar null, mesmo não havendo necessidade de passar nenhum parâmetro.
Ao fazer a requisição o objeto vai executar o método on*readyState*change.
Esse código vai ser executado várias vezes durante a requisição, por isso é testado*readyState*. Quando *readyState* tiver

o valor 4, significa que a requisição foi concluída e que é possível ler o retorno e trabalhar com ele.
Para capturar a resposta do documento web acessado, você tem duas propriedades do objeto *XMLHttpRequest*: *responseText* e *responseXML*.
A propriedade *responseText* contém o retorno do documento web acessado na forma de texto. Já a propriedade *responseXML* retorna um objeto DOM, em formato XML, podendo ser manipulado facilmente.