Article
Home> Articles > AJAX > Basics

Learn AJAX Tutorial

This AJAX tutorial will help you to learn AJAX using AJAX source code snippets and diagrams.

Introduction to AJAX

Foreword:

This tutorial provides an introduction to AJAX; initially by explaining the different associated concepts and then presenting how to practically use AJAX in simple applications.

Basic Requirements:

You should have some basic knowledge about HTML/XTML and Javascript before you start with AJAX .

What is AJAX – Asynchronous JavaScript And XML?

AJAX is simply about JavaScript on the client (web browser) communicating with the server using XML. It is based on JavaScript and HTTP requests.

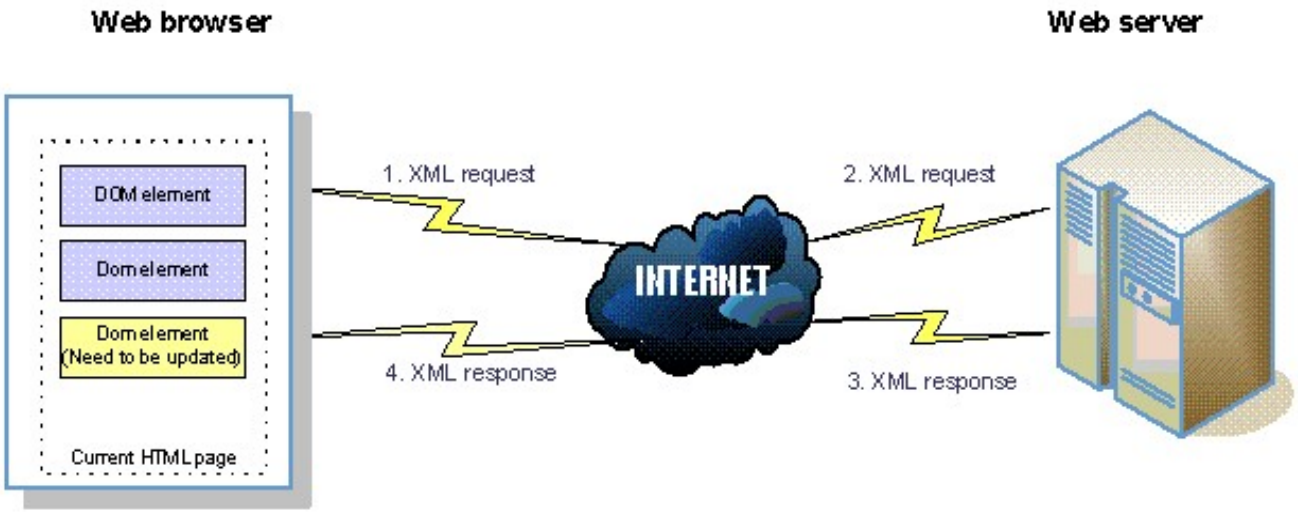
By using AJAX,better,faster and easier to use web applications can be created.

AJAX is not a new programming language it just introduces a new method of using the existing language standards.

AJAX web model

Ajax Web Model

The Ajax web model is based on the traditional web model with changes to the transmitted message on the web server and the web browser. The following diagram presents the AJAX web model.



In the figure,the current HTML page displayed on the web browser has three parts. The two purple parts don’t need to be updated; only the yellow part needs to be.

Submitting a Request

What happens when the user submits a request?

When user submits request,the steps below occur sequentially:

1. Web browser requests for the content of just the part of the page that it needs.

2. Web server analyzes the received request and builds up an XML message which is then sent back to the web browser.

3. After the web browser receives the XML message,it parses the message in order to update the content of that part of the page.

AJAX uses JavaScript language through HTTP protocol to send/receive XML messages asynchronously to/from web server.

Asynchronously means that data is not sent in a sequence.

Benefits of AJAX

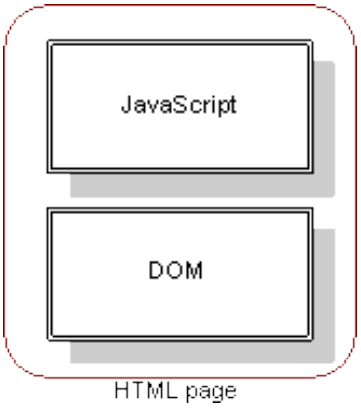
Benefits:

AJAX brings to web applications new characteristics that are not available as standard in traditional web applications.

- Continuous uploading,time is not wasted while waiting for page redraws and reloads
- Real-time updates,certain parts of the web pages are directly updated without requiring the whole page to updated.
- Greater graphical interaction similar to desktop applications – drag and drop
- Standard mechanism for the client to interact with the server and only send small xml fragments.

Role of JavaScript

An HTML page has two components as shown in the figure below:



Document Object Model (DOM):
• The W3C Document Object Model is a platform and language neutral interface that allows programs and scripts to dynamically access and update content,structure and style of a document

• The HTML DOM is the Document Object Model for HTML. The HTML DOM defines a standard set of objects for HTML and a standard way to access and manipulate HTML documents

• The HTML DOM views HTML documents as a tree structure of elements. All elements,along with their text and attributes,can be accessed and manipulated through the DOM tree

• The HTML code below shows DOM Body,Form and Input objects. The Input objects are embedded in the Form and the Form is embedded in the Body object

```
 1: <html>
 2:  <head></head>
 3:
 4:  <body>
 5:   <form id=”loginForm”>
 6:    <input type=”text” id=”userName”/>
 7:    <input type=”submit” id=”login” value=”Login”/>
 8:   </form>
 9:  </body>
10: </html>
```

The languages that can be used within the HTML DOM to access DOM objects are Java,JavaScript,and VBScript

Java Script in relation to HTML

Javascript

- JavaScript is a scripting language that is supported and runs on almost all web browsers,such as Internet Explorer,Mozilla,Firefox,Nestcape and Opera.
- JavaScript was designed to add interactivity to HTML pages
- By embedding JavaScript in an HTML page,parts of the Document Object Model (DOM) within the HTML document can be updated.

Example:
A JavaScript function will be added into the HTML code above to handle the *onClick* event of the Login button. As the user clicks the button,the notification message:*"The name is required field and can not be empty"* will occur,if the user did not type in his name. The code as in following:

```
 1: <html>
2:         <head>
3:                     <script type="text/javascript">
4:
5:                     function validateUser()
6:                     {
7:                                 name= document.getElementById("userName").value;
8:                                 if (name == "")
9:                                 {
10:                                 alert("The name is required field and can not be empty");
11:                                 }
12:                    }
13:
14:                    </script>
15: </head>
16:
17: <body>
18:
19: <form id="loginForm">
```

```
20: <input type="text" id="userName"/>
21: <input type="submit" id="login" value="Login" onClick="validateUser()"/>
22: </form>
23:
24: </body>
25: </html>
26:
```

How does AJAX work

Ways to submit a request to the web server

In the traditional web model,HTML Form elements are used to submit request to web server. The HTML code below is an example of an HTML form:

```
1: <form id=”loginForm” action=”login.jsp”>
2: <input type=”text” id=”userName”/>
3: <input type=”submit” id=”login” value=”Login” >
4: </form>
```

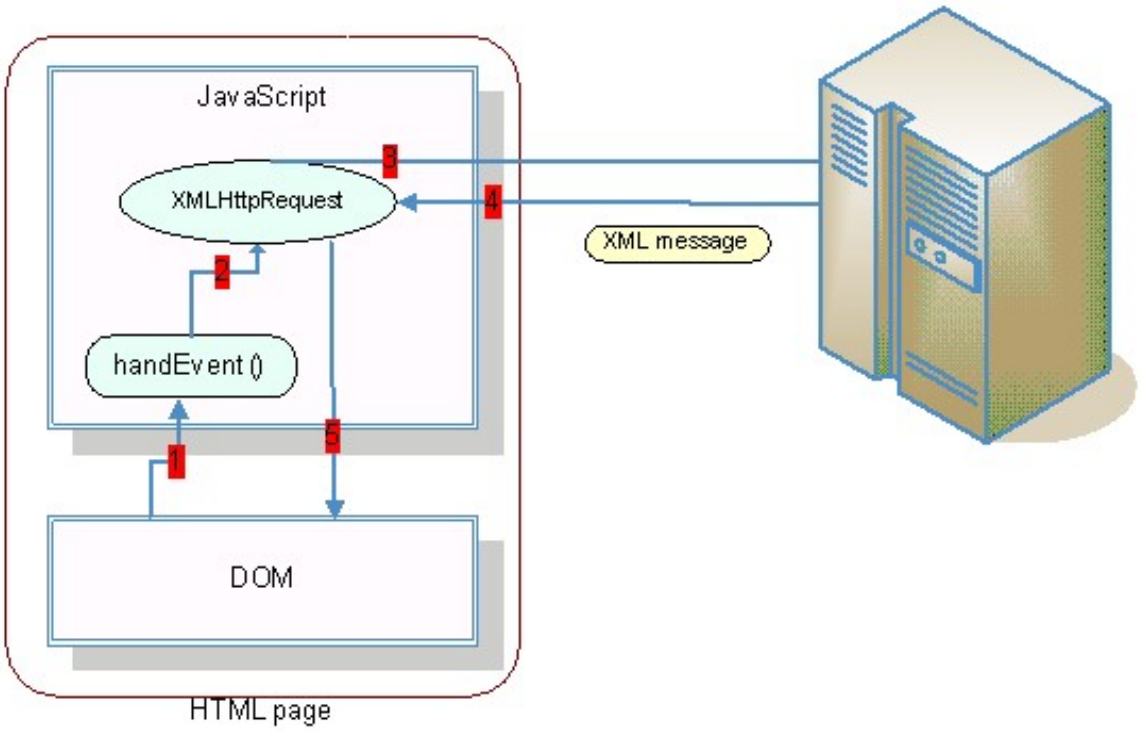
The data of the sent message is combined with the data in the html form. In the above code,the content of the message is the value of the userName textbox.

After submitting the request,the web browser then receives a new HTML page from the web server

In the Ajax web model,there is another way,by using the XMLHttpMessage object to communicate with web server. The XMLHttpMessage object is used to send/receive messages to and from the web server.

Some common steps that Ajax application follows:

The figure below describes the structure of HTML pages and a sequence of actions in Ajax web application.:



Notice the three elements in the HTML page:

· The handEvent() JavaScript method,this method is used to handle events of a special HTML element

· XMLHttpRequest is an object supported by the web browser and runs as web client

· XML message - message transferred between XMLHttpRequest object and web server is under XML format

Ajax web application follows the sequential steps below; the actions are also mapped from the above figure:

1. The JavaScript function handEvent()t will be invoked when having an event occurred on the HTML element.

2. In the handEvent() method,an instance of XMLHttpRequest object is created.

3. The XMLHttpRequest object organizes an xml message within about the status of the HTML page,and then sends it to the web server.

4. After sending the request,the XMLHttpRequest object listens to the message from the web server

5. The XMLHttpRequest object parses the message returned from the web server and updates it into the DOM object

For a demonstration of the steps,a code example is presented in the next section.

Examples of AJAX

Below are two simple examples for using Ajax . One is an example for sending and getting a text message. The other one is an example for parsing an XML message

Click here to download the example about validating a user name. A text message is used in this example.

Click here to download the example about car information. XML message is used in this example.

To see how to run the demonstrations, look at the video here to find out how the Ajax example runs on different web browsers.

Click here to download the demonstration for how an Ajax web application runs.

Code Example for creating an AJAX application

In this section,an example for Ajax web application is created step by step. The example is based on a simple login page that validates the username entered in.

Requirements:

· The web application has a Login page containing username and password textboxes

· When the user types in his name, he will receive notifications about the status of the current value in the username textbox

· If the current value is a beginning part of the string “VisualBuilder”,the user will receive a green notification “Continue..”

· If the current value is not a beginning part of the string,user will receive a red notification - “Invalid input name”

· If the current value is equal to the string, user will receive a blue notification “Valid input name”

This web application includes only a web page and a service running on web server.

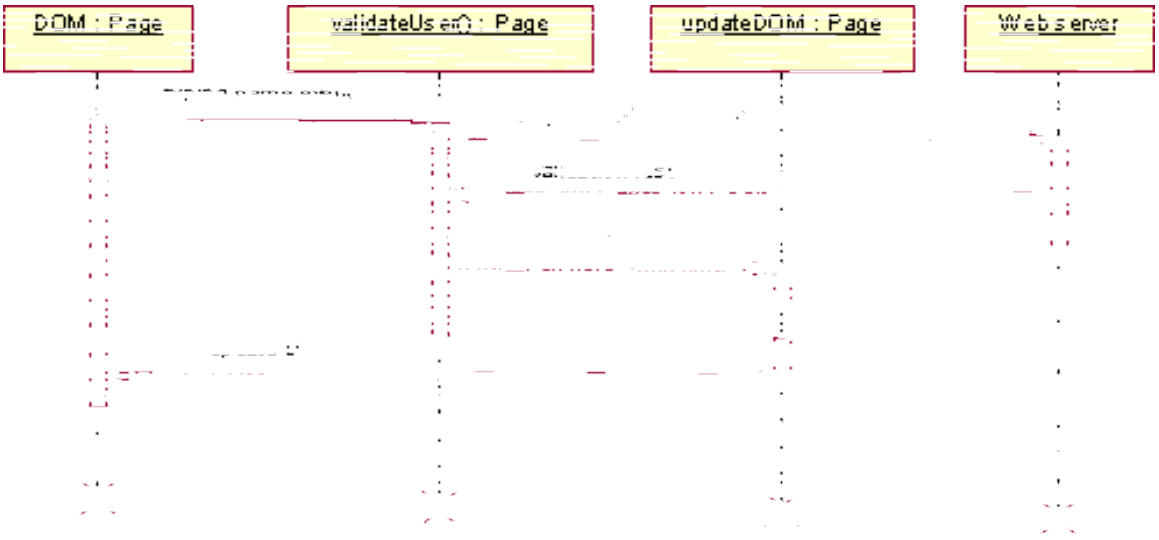
The web page is HTML code with the embedded JavaScript functions:

· ValidateUser() to catch the user’s typing event on the username textbox and to send/receive the XML message

· UpdateDOM() to parse the received message and update the status on the HTML page.

The service running on the web server has the role to listen to the request from the web browser then send back an appropriate XML response.

The diagram below describes the steps above in a sequence of methods invoked:



There are three key points in creating an Ajax application,which are also applicable to this example:

• Using XMLHttpRequest object to send xml message to web server
• Create a service that runs on web server to respond to request
• Parsing XMLHttpRequest object then update to DOM object of the html page on client-side

Using XMLHttpRequest object to send xml message

Using XMLHttpRequest object to send xml message

Firstly,the login page is presented (login.jsp) where the user types and receives the status immediately afterwards.

When the user is typing in the username textbox,the onKeyUp event is sent and the JavaScript function validateUse() will be invoked,as written in following:

```
<input type=text id=user_name onKeyUp=validateUser()>
```

In the validateUser method,an instance of XMLHttpRequest will be created:

```
if (window.ActiveXObject)
{
 oXMLRequest = new ActiveXObject(Microsoft.XMLHTTP);
}
else
{
oXMLRequest = new XMLHttpRequest();
}
```

There are two ways of creating an instance of XMLHttpRequest object. They deal with the different kinds of browser,one for Internet Explorer and one for the other browsers such as Mozilla,Safari.

The instance of XMLHttpRequest object is used to send the XML request,and receive the response. Before sending,it must be known where the service on web server located the message and the sending method. The code below is an example for setting up the request:

```
var strUserName = document.getElementById(user_name).value;
var strValidationServiceUrl = user.UserValidator?UserName= + strUserName;

oXMLRequest.open(GET,strValidationServiceUrl,true);
oXMLRequest.onreadystatechange = updateDOM;
oXMLRequest.send(null);
```

1. **The Open Method:**
  The open method is to identify:

- The URL of destination service running on web server,
- Sending method is the http method,typical value is GET or POST but can also be HEAD
- Boolean value identifies that communication between XMLHttpRequest and web server is asynchronous or not

The property onreadystatechange is to specify the method to update the DOM object of the HTML page. This method will be invoked immediately after receiving the response from the web server. In the code above it’s the updateDOM.

2. **The Send Method:**
The send method is used to send a request to the service specified in URL,value of input parameter is content of the request. In this case,the URL parameter is used to tell the web server what is required instead of the XML message,so the

XML message is set as null.

Create service responding to http request

Create service responding to http request:

Secondly,on the server-side,within Java web application,we use one Java Servlet as service running on web server. This servlet is to validate user name. It has the doGet method to service http GET method:

```
public void doGet(HttpServletRequest oRequest,HttpServletResponse oResponse)
throws IOException,ServerException
{
String strUserName = oRequest.getParameter(“UserName”);

oResponse.setContentType(text/xml);
oResponse.setHeader(Cache-Control,no-cache);
if (“VisualBuilder”. indexOf(strUserName) == 0)
{
        if (“VisualBuilder”.equals(strUserName))
        {
        oResponse.getWriter().write(“Valid”);
        }
        else
        {
        oResponse.getWriter().write(“Continue”);
        }
}
else
{
        oResponse.getWriter().write(“Invalid”);
}
}
```

The doGet checks user name posted by the client. If the is used name as a beginning part of “VisualBuilder”,the value of text response will be “Continue”,this means user is typing correctly. If the name is equal to “VisualBuilder”,the response will be “Valid”. However,in other cases,the response will be “Invalid”. At the last,the doGet sends back this response to client.

Parsing xml response and update DOM object

Parsing xml response and update DOM object

Eventually,after receiving the response,the DOM object of the client html page will be updated. We do that in the updateDOM method:

```
function updateDOM()
{
if (oXMLRequest.readyState  == 4)
        {
        if (oXMLRequest.status  == 200)
        {
                strStatus = oXMLRequest.responseText;

                 var statusDiv = document.getElementById(status);
        if (document.all)
                {
                        statusDiv = document.all[status];
                 }

                 if (strStatus == Valid)
                 {
statusDiv.innerHTML = <div style='color:green'> Valid user</div>;
                  }
                  else if (strStatus == Invalid)
                  {
statusDiv.innerHTML = <div style='color:red'> Invalid user</div>;
}
                  else if (strStatus == Continue)
                  {
statusDiv.innerHTML = <div style='color:green'> Continue..</div>;
                  }
         }
}
```

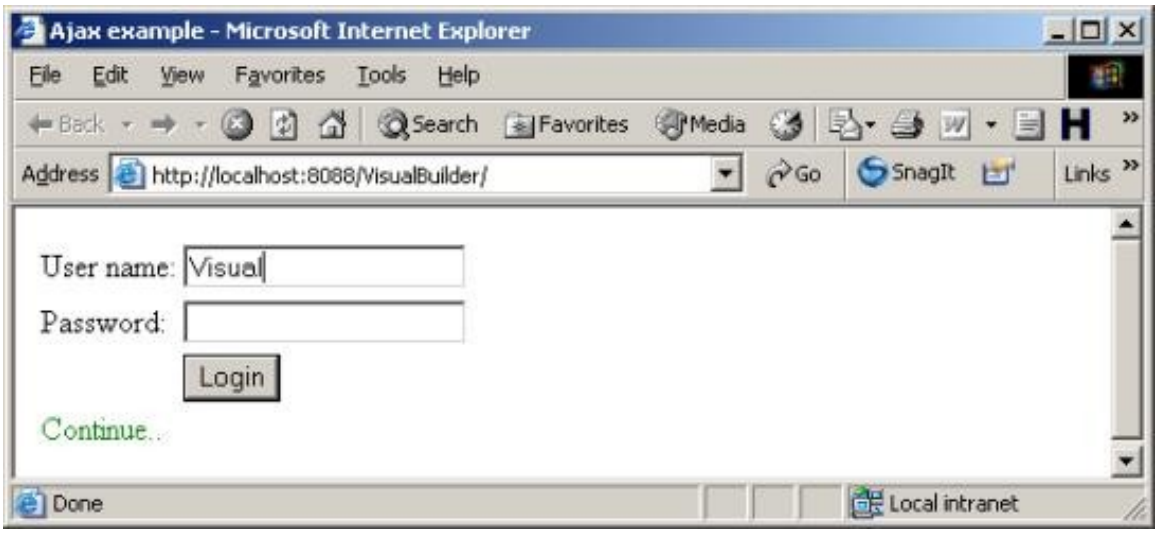
The conditional line: oXMLRequest.readyState = 4 tells us that the XMLHttpRequest instance have just received the response message and the line oXMLRequest.status=200 which means that the service for response message has been done successfully.

The response message can be under text or xml format. In this method,parsing xml message is very simple: by getting the text value of the responseText property. Still there is another way to get content of response,by parsing value of the responseXML property. You could refer to the Ajax example for parsing xml for it.

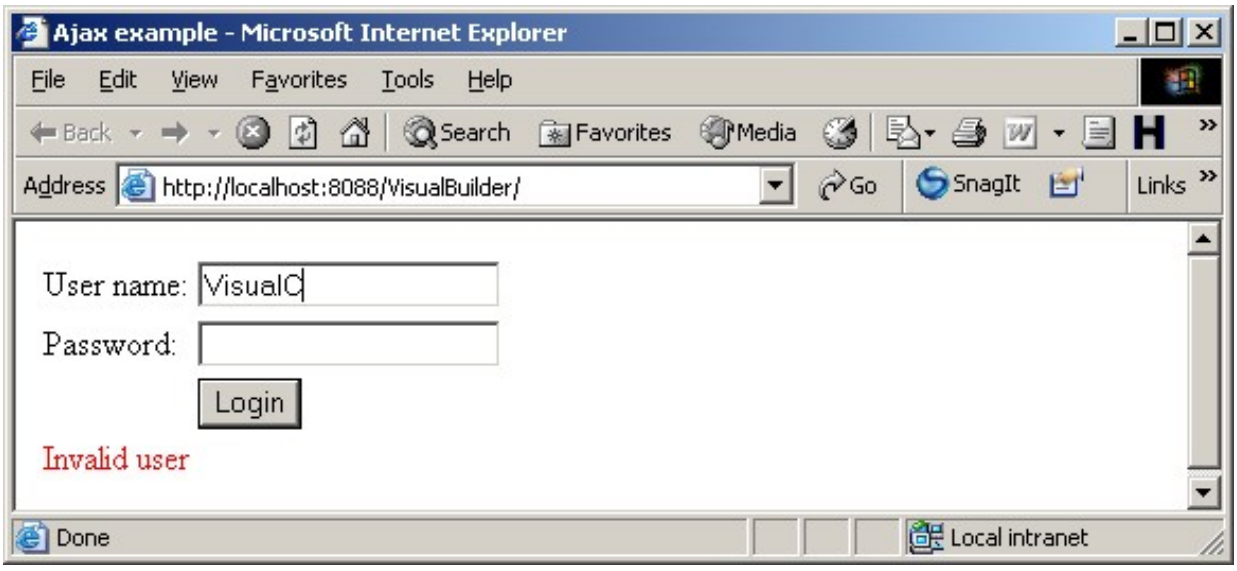
Screen shots of running an AJAX example

Example:

The following are screenshots of running the example:
Start by entering “Visual” for VisualBuilder



If “VisualC” is entered an invalid user message is displayed:



To get a valid user entered type'VisualBuilder':

To download the login.jsp page,click here

To download the Java code of the ValidateServlet,click here

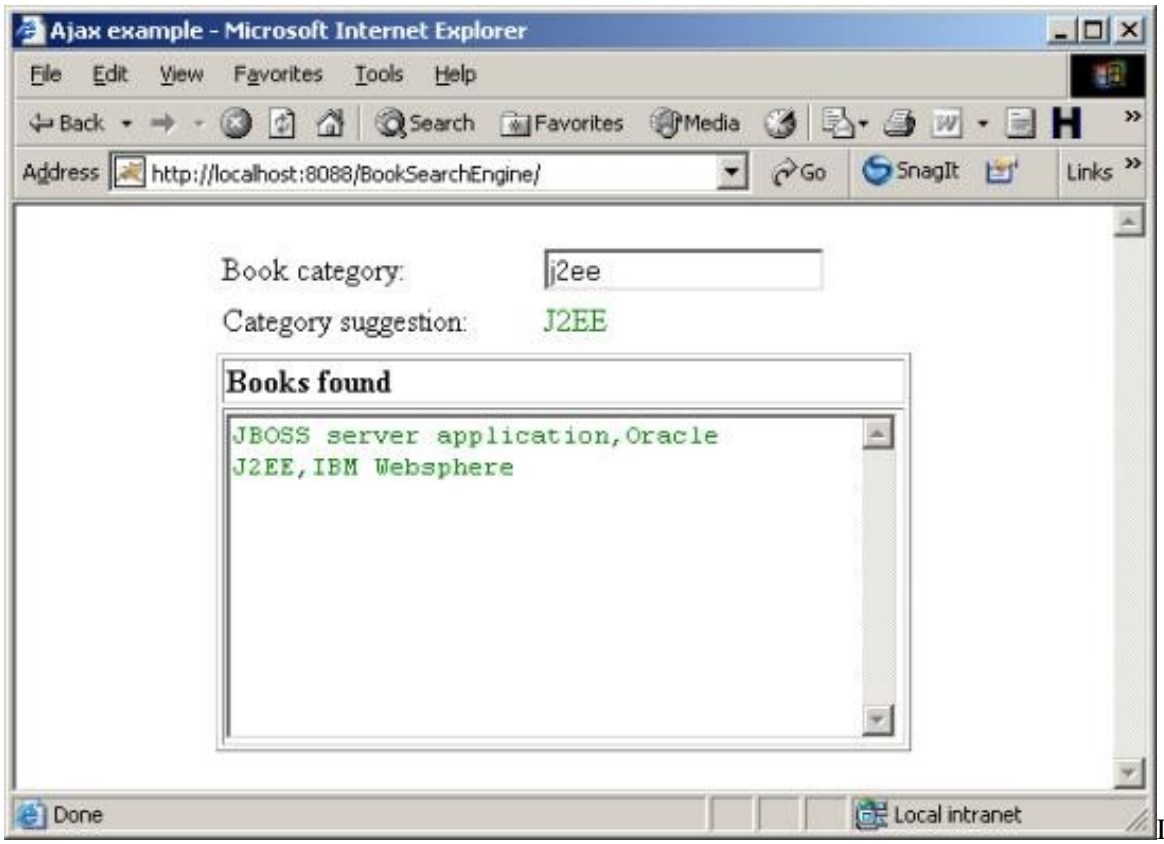
Example for parsing XML message

Example for parsing XML message

This web example shows how to search books. When the user types into the category textbox,a list of book categories is received that have names near to value of the category textbox. If the value of the category textbox is found,the books textbox will display all books of this category

The screen below shows:

- User types category: j2ee
- He receives a suggestion: J2EE
- J2EE exists,the books textbox displays all books of the category including:Jboss server application,Oracle J2EE,IBM Websphere



In this example, we used HTTP messages between the web browser and web server as XML messages

There are three key points in using with:

- In client-side,build and send XML message
- In server-side,parse received XML message and build to respond a new XML message
- In client-side,parse the response XML message and update to DOM

Building response XML message in server

Building response XML message in server

A Java Servlet is used to check the request from the client as in the first example. But this servlet is used in the POST method instead of the GET method.

The doPost() method of the servlet is to examine request in POST HTTP method from client,it looks like:

```
public void doPost(HttpServletRequest oRequest,HttpServletResponse oResponse) throws IOException,ServerException
        {
                // Get category name passed by client
                String strCategory = oRequest.getParameter(category)

                // Get output writer
                PrintWriter oWriter = oResponse.getWriter();

                // Set response in XML format
                oResponse.setContentType(text/xml);
                oResponse.setHeader(Cache-Control,no-cache);

                 // Out XML message to client
                 oWriter.print(getXMLResponseMessage(strCategory));
        }
```

This code shows that when each POST request is caught,the servlet parses to get value of the category passed by the client and will look up for the list of books based on the given category,then respond it.

With XML message,we notice in:

·Set response message as XML format instead of TEXT as in the first example
·Build response XML message

The getXMLResponseMessage(strCategory) method is to build XML message containing a suggestion and a list of related books. It returns the message with a structure as below. In there,the suggestion node contains suggested categories and the books node contains a list of books it has.

```
<book_response>
        <suggestion></suggestion>
        <books>
                <book></book>
                <book></book>
                <book></book>
         </books>

</book_response>
```

Parsing XML message and update to DOM

Parsing XML message and update to DOM:

Now,after receiving XML message,we need to parse it to get suggestions and the book list and then update to DOM. The callback JavaScript updateBooks() will do this. Its code includes two parts: update suggestion and books

The section code below is to get the root document of the XML message:

```
// Get XML response message
var XMLresponse = oXMLRequest.responseXML;

// Point to the 'book_response' node
var docRoot = XMLresponse.getElementsByTagName(book_response)[0];
```

In order to get and update suggestion using the XML structure shown above,we do as:

```
 // Get data of the Suggestion node
 var strSuggestion = docRoot.childNodes[0].firstChild.data;

 // Get a HTML DOM object that need to be updated
 var suggestionHTMLDiv = document.getElementById(suggestion);
 f (document.all)
 {
        suggestionDiv = document.all[suggestion];
  }

 // Display the received suggestion
suggestionHTMLDiv.innerHTML = <div> strSuggestion </div>;
```

We update the book list similar to the given suggestion. However,we should notice the way to get data from the tree node.

```
// Get the books HTML DOM object
var bookHTMLText = document.getElementById(books);
if (document.all)
{
        bookHTMLText = document.all[books];
 }

 // Point to the Books node
 var oBooks = docRoot.childNodes[1];

 // Parse XML message to build a book list
 var strBooks = ;
 for (var i = 0; i < oBooks.childNodes.length; i )
 {
        if (strBooks == )
        {
              strBooks = oBooks.childNodes[i].firstChild.data;
        }
```

```
        else
        {
                strBooks = strBooks , oBooks.childNodes[i].firstChild.data;
         }
}

// Display the books list
bookHTMLText.value = strBooks;
```

For demonstration of running this application,you could refer to the Example of Ajax section

XmlHttpRequest object,how to use?

XmlHttpRequest object,how should I use?

XMLHttpRequest object is a kind of web client that can retrieve and submit XML data directly then convert retrieved XML data into HTML content.

XMLHttpRequest object is supported in almost browser providers:

- Microsoft,it works as an ActiveX and is supported in Internet Explorer 5.0 or later as an ActiveX object

- Mozilla,in Firefox1.0 or later

- Apple,in Safari1.2 or later

How to create?
Currently,there are two ways to create object based on the browser provider:

For Microsoft:

Var oXMLRequest = new ActiveXObject(“Microsoft.XMLHTTP”);

For Mozilla and Safari

Var oXMLRequest = new XMLHttpRequest();

Object methods

Followings are the methods in the XMLHttpRequest object supported by Safari1.2,Mozilla and Windows IE 5.0 or later

Method	Description
abort()	Stops the current request
getAllResponseHeaders()	Returns complete set of headers (labels and values) as string
getResponseHeader(“headerLabel”)	Returns string value of a single header label
open(“method”,“URL”,[. asyncFlag[,”userName”[,”password”]]])	This method is used to sets up a new request to server. Its parameters specify destination URL,sending method and other optional attributes of a pending request Sending method is either Post or Get
send(content)	Transmit the request,optionally with post-able string or DOM object data
setRequestHeader(“label”,“value”)	Assign a label/value pair to the header to be sent with a request

Object properties of XMLHttpRequest object

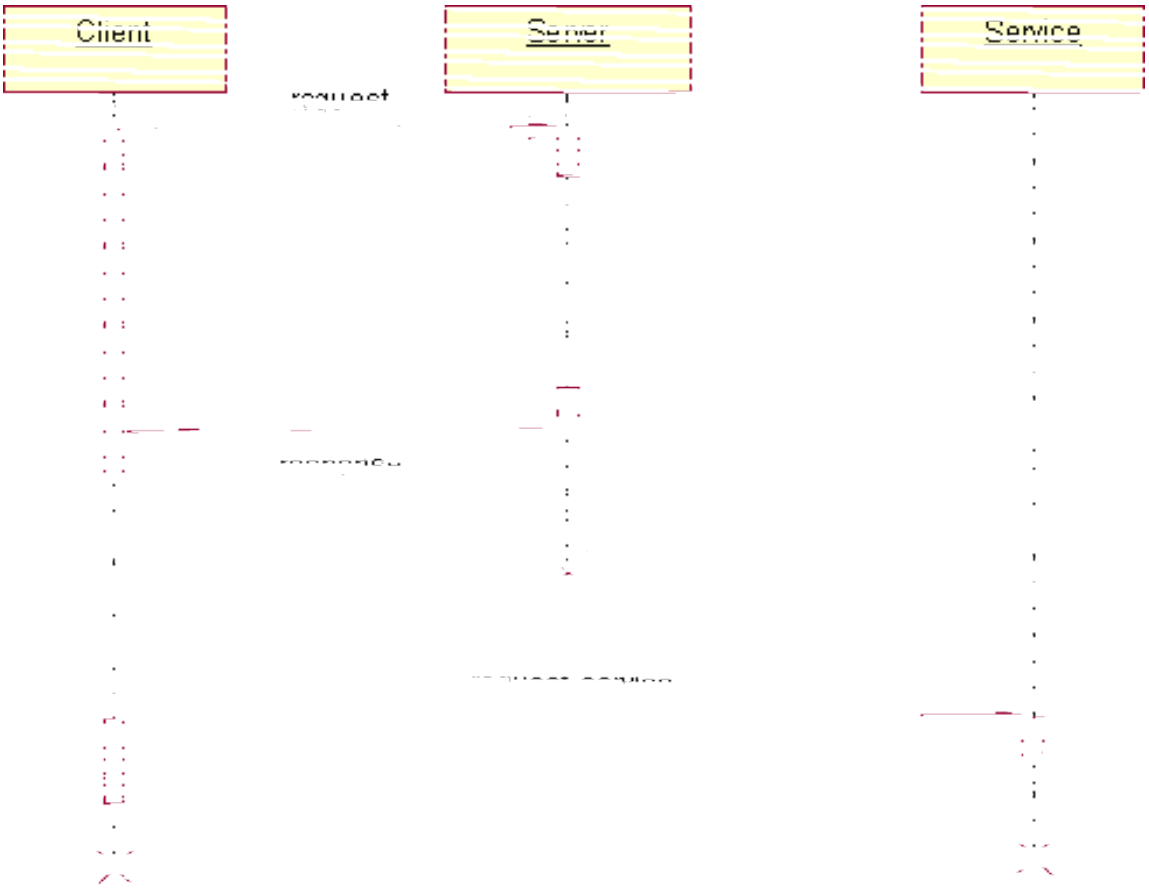
Following is the table of all properties of *XMLHttpRequest* object:

Property	Description
onreadystatechange	Event handler for an event that fires at every state change
ReadyState	Object status integer: 0 = un-initialized 1 = loading 2 = loaded 3 = interactive 4 = complete
responseText	The text that server sends back to respond to a request.
responseXML	DOM-compatible document object of data returned from server
Status	Numeric code returned by server 404 = “Not Found” 200 = “Ok”
StatusText	String message accompanying the status code

Make asynchronous requests with JavaScript and AJAX

There are two ways to communicate with the client and the server; synchronous and asynchronous. Let’s take a look at them:

The sequence diagram below is an example of the synchronous communication model:



In this model,the client (browser or client application) makes a request to the server then waits until it receives a response from the server before doing anything.

This process consumes unnecessary time.

Suitability of the synchronous and asynchronous models

Suitability:

The synchronous model is not suitable with the kind of client application that often does interactions to the server. Especially within a web application because it makes them very slow.

With asynchronous model,the client application does not consume much of the user's time making him wait for the response. Therefore,the user feels in real-time while interacting. The sequence diagram below shows the functioning of an asynchronous model:

This diagram shows that after sending the request,the client can do something else instead of wasting his time waiting for the response and doing nothing.

XMLHttpRequest object supports the asynchronous model. Using it in web application does not make you wait for the server to response after making a request. The web application continues with other tasks rather than staying and listening to the response. Therefore,it prevents users from spending a long time waiting.

XMLHttpRequest object has the open method to set asynchronous or not,like as in following:

```
var strValidationServiceUrl = "userValidation?user=VisualBulder";
var bAsynchronous = true;
var strHttpMethod = "GET";

oXMLRequest.open(strHttpMethod,strValidationServiceUrl,bAsynchronous);
```

AJAX Frameworks

AJAX Frameworks:

There are many frameworks for Ajax,like JSF and Struts frameworks for Servlet/JSP,developed to help developers more clearly and easily to write an Ajax project. In this section we will review several remarkable Ajax frameworks:

- Ajax JSP Tag Library
- Ajax.Net
- OpenRico
- Prototype
- Sarissa

Ajax JSP Tag Library:
The Ajax Tag Library is a set of JSP tags that simplifies the use of Asynchronous JavaScript and XML (AJAX) in Java Server Pages (JSP).

For more details,please refer to http://ajaxtags.sourceforge.net/

Ajax.Net:
Ajax.Net is a library enabling various kinds of access from JavaScript to server-side.Net. It is able to access session data from JavaScript without source code changed on server-side. It also provides full class support for the returned values on the client-site's JavaScript including DataTable,DataSet,DataView,Arrays and Collections

For more details,refer to http://weblogs.asp.net/mschwarz/

OpenRico:
Rico provides a very simple interface for registering Ajax request handlers as well as HTML elements or JavaScript objects as response objects. It also provides interfaces for enabling web application to support drag and drop..

For more details,refer to http://openrico.org/

Prototype
Prototype is a JavaScript framework including a solid Ajax library and a toolkit to simplify its use.

For more details,refer to http://prototype.conio.net/

Sarissa
Sarissa is a JavaScript API that summarizes XML functionality in browser-independent calls. It supports a variety of XML technologies,including XPath queries,XSLT,and serialization of JavaScript objects to XML

For more details,refer to http://sarissa.sourceforge.net

Date entered : 8th Feb 2007

Rating :*No Rating*

Submitted by : visualbuilder

Document created date: 29 Aug 2007