Pratical – AJAX on Java Web Application

Objectives:

- Integrates AJAX, Javascript with Java Servlet

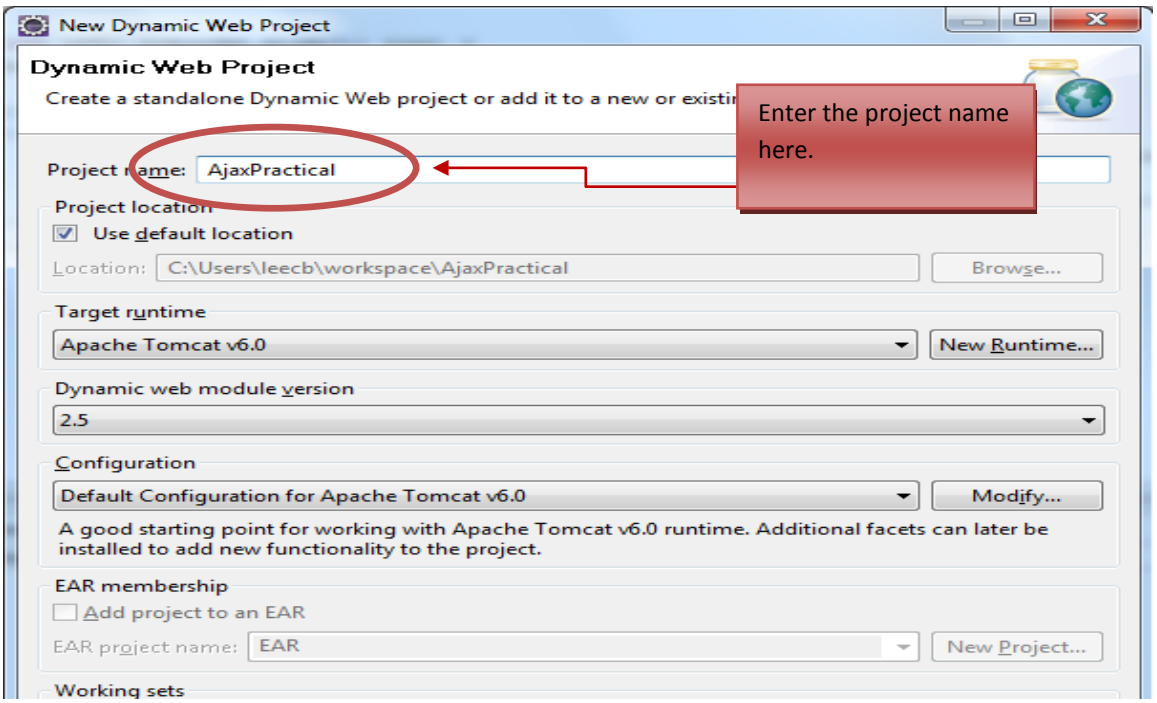
This practical consist of 4 parts :

1) Java Servlet without AJAX (Traditional)
2) Java Servlet with AJAX enabled
3) Java Servlet with AJAX enabled and MySQL
4) Java Servlet with AJAX enabled , MySQL & XML
5) ~~*(optional) Java Servlet with AJAX,MySQL, XML & AutoComplete*~~

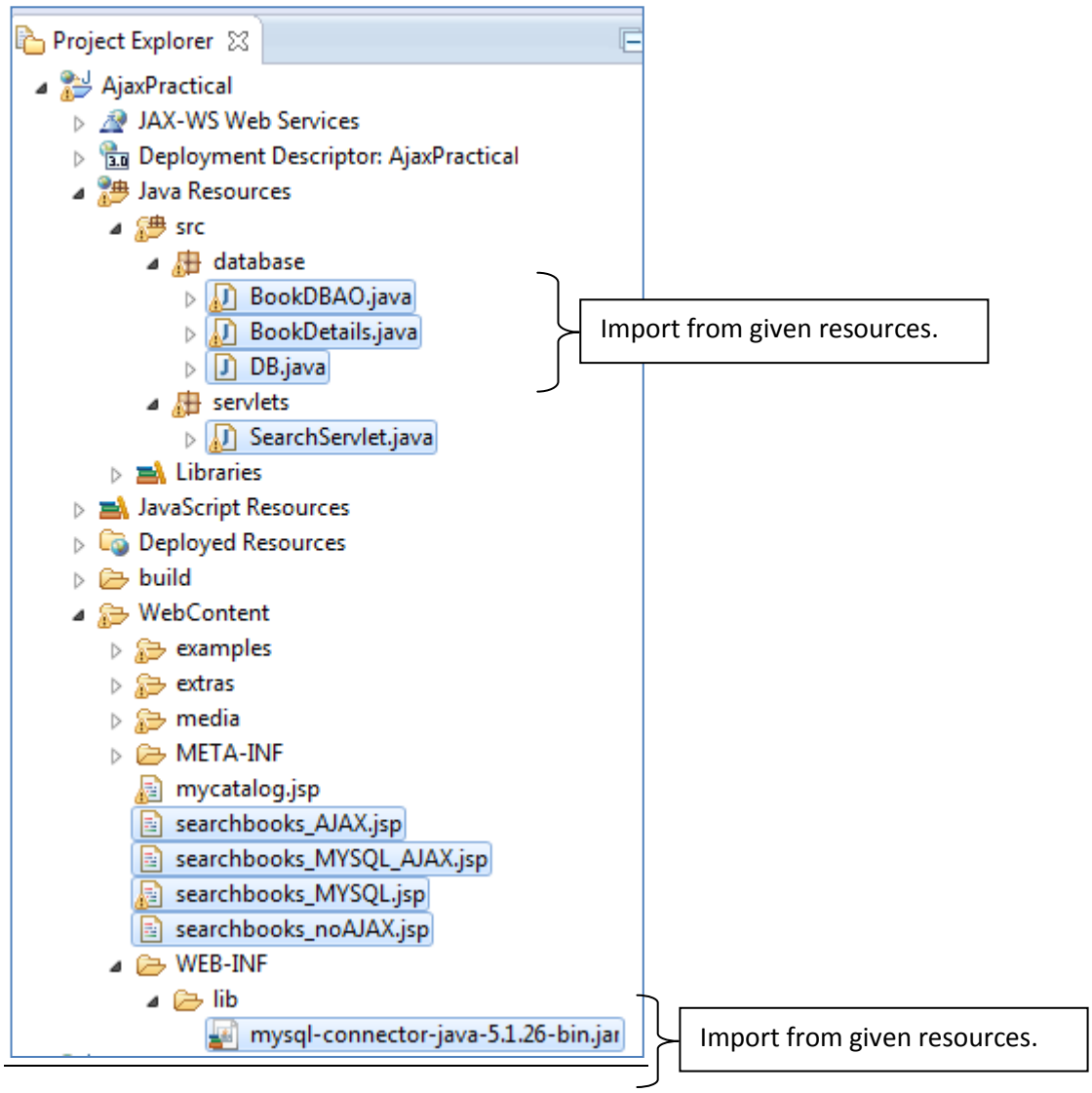
Create Dynamic Web Project - AjaxPractical

To get started with DataTables, download **P11_Resources.zip** from our resource site. Extract the files. Make sure that the database is loaded with the Bookstore tables that we have been using for the past few weeks

Starts your Eclipse, and verify that your Tomcat Server is created, and MYSQL database is started correctly (Take note of the port that MYSQL is configured). Creates a **Dynmaic Web Project** with information given below, click **Finish** upon completion.



Workspace

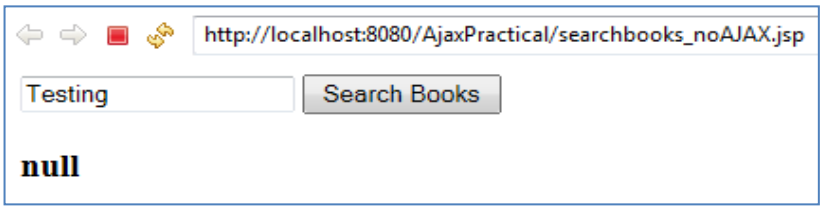

Project Explorer
AjaxPractical
JAX-WS Web Services
Deployment Descriptor: AjaxPractical
Java Resources
src
database
BookDBAO.java
BookDetails.java
DB.java
servlets
SearchServlet.java
Libraries
JavaScript Resources
Deployed Resources
build
WebContent
examples
extras
media
META-INF
mycatalog.jsp
searchbooks_AJAX.jsp
searchbooks_MYSQL_AJAX.jsp
searchbooks_MYSQL.jsp
searchbooks_noAJAX.jsp
WEB-INF
lib
mysql-connector-java-5.1.26-bin.jar
Import from given resources.
Import from given resources.

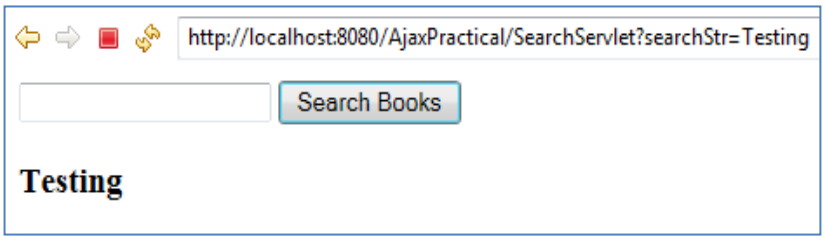
1. A Book Store (Search Feature) Java Servlet (Traditional)

Create a JSP and Servlet that provides a search feature to an online bookstore. For a start, the Servlet will just return the search string that the user has entered. Create a Java-based web application with the following files :

JSP page	: **searchBooks_noAJAX.jsp** (inside **WebContent**)
Servlet	: **SearchServlet.java** (inside package **servlets**)

Screens





JSP Codes :

```
<html>
<head>
<title>Search</title>
</head>

<body>
<form name="SearchForm" action="SearchServlet" method="get">
<input type="text" size="20" id="searchStr" name="searchStr">
<input id="submit_btn" type="Submit" value="Search Books">
</form>

<%
  String book1 = (String) request.getAttribute("searchStr");
%>

<div id="myDiv">
<h3><%=book1 %></h3>
</div>
</body>
</html>
```

Servlet Codes :

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

	response.setContentType("text/html");
	PrintWriter out = response.getWriter();
	String str = request.getParameter("searchStr");

	request.setAttribute("searchStr", str);
	request.getRequestDispatcher("/searchbooks_noAJAX.jsp").forward(request,
	response);
}
```

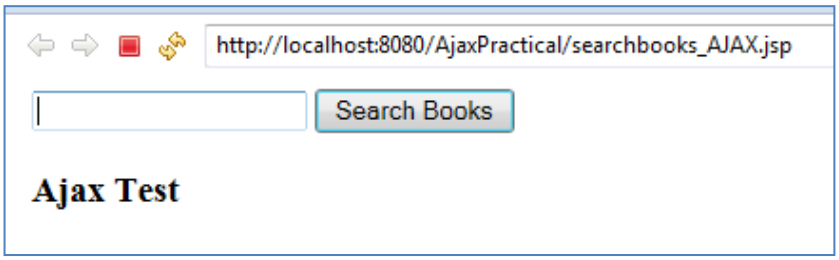
2. An AJAX-enabled Web Application

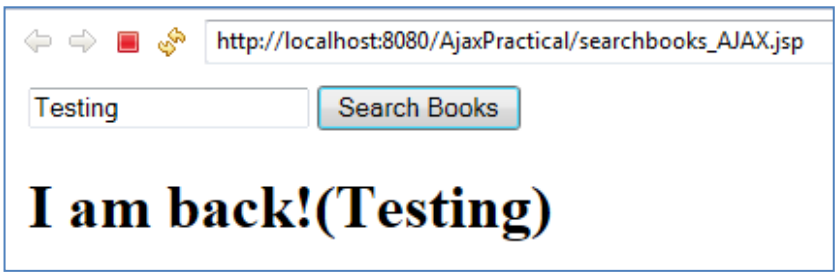
AJAX is an efficient way for a web application to handle user interactions with a web page -- a way that reduces the need to do a page refresh or full page reload for every user interaction. This enables rich behavior (similar to that of a desktop application or plugin-based web application) using a browser. AJAX interactions are handled asynchronously in the background. As this happens, a user can continue working with the page.

Create a new JSP and Servlet that provides a search feature to an online bookstore with AJAX enabled. For a start, the Servlet will just return the search string that the user has entered. Create a Java-based web application with the following files :

JSP page	**: searchBooks_AJAX.jsp** (inside **WebContent**)
Servlet	**: SearchServlet.java** (inside package **servlets**) ** use the same SearchServlet or create a new one.

Screens





JSP Codes :

```
<html>
<head>
<title>Search</title>

<script>
 function initRequest() {
       if (window.XMLHttpRequest) {
           return new XMLHttpRequest();
       } else if (window.ActiveXObject) {
           isIE = true;
           return new ActiveXObject("Microsoft.XMLHTTP");
       }
 }

 function sendRequest() {
        var a = document.getElementById('searchStr');
        var url = "SearchServlet?searchStr=" + escape(a.value);
        var req = initRequest();
        req.onreadystatechange = function() {
            if (req.readyState == 4) {
              if (req.status == 200) {
                document.getElementById("myDiv").innerHTML =  req.responseText;
              }
            }
        };
        req.open("GET", url, true);
        req.send(null);
 }
</script>
</head>

<body>
<input type="text" size="20" id="searchStr" name="searchStr">
<input id="submit_btn" type="Submit" value="Search Books" onClick="sendRequest();">

<div id="myDiv">
<h3>Ajax Test</h3>
</div>

</body>
</html>
```

*** Note : <form> is not required.*

Servlet Codes :

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

	response.setContentType("text/html");
	PrintWriter out = response.getWriter();
	String str = request.getParameter("searchStr");

	out.println("<h1>I am back!(" + str +")</h1>" );

}
```

3. An Non-AJAX Enabled Web Application with MYSQL

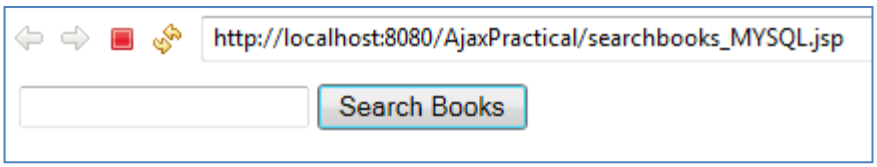
Create a JSP and Servlet that provides a search feature to an online bookstore with MYSQL as datastore. The Servlet extract data from Database based on the searchStr and return data as a List of 'BookDetails' objects. Create a Java-based web application with the following files :

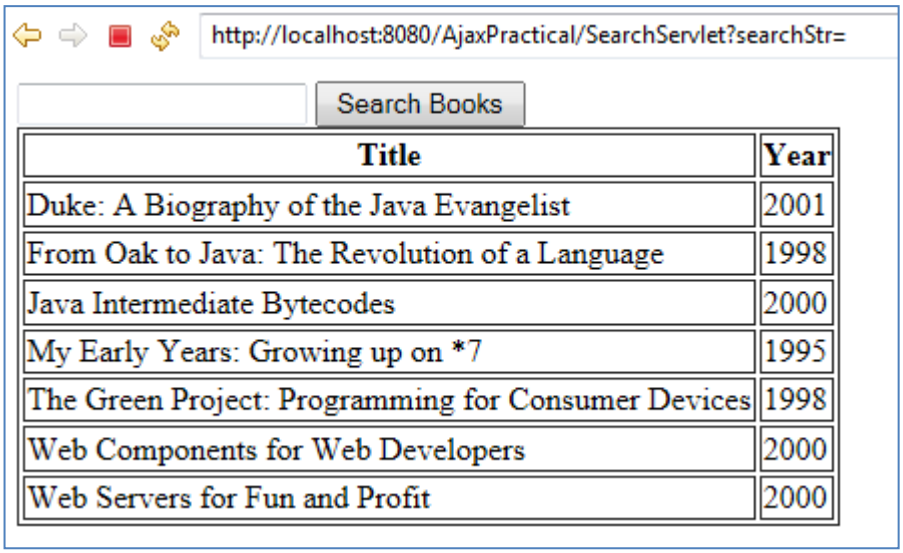
JSP page	**: searchBooks_MYSQL.jsp** (inside **WebContent**)
Servlet	**: SearchServlet.java** (inside package **servlets**) Response : Objects List <BookDetails> ** use the same SearchServlet or create a new one.
Database	**MYSQL – 'test' database** ** import database into your project

Screens

To return all results – leave the search string as empty

A total 7 books will be returned.





Title	Year
Duke: A Biography of the Java Evangelist	2001
From Oak to Java: The Revolution of a Language	1998
Java Intermediate Bytecodes	2000
My Early Years: Growing up on *7	1995
The Green Project: Programming for Consumer Devices	1998
Web Components for Web Developers	2000
Web Servers for Fun and Profit	2000

JSP Codes :

```
<html>
<head>
<title>Insert title here</title>
</head>

<%@ page import="database.*, java.util.*" %>
<body>
<form name="SearchForm" action="SearchServlet" method="get">
<input type="text" size="20" id="searchStr" name="searchStr">
<input id="submit_btn" type="Submit" value="Search Books">


	<%
		List <BookDetails> bookList = null;
		bookList = (List)request.getAttribute("books");

		if(bookList != null){

	%>
			<table class='display' id='table' border='1'>
				<thead>
				  <tr>
				  <th>Title</th>
				  <th>Year</th>
				  </tr>
				</thead>
			<tbody>

	<%
			for (int i=0; i<bookList.size(); i++) {
				BookDetails book = (BookDetails)bookList.get(i);
	%>

				  <tr>
				      <td><%= book.getTitle() %></td>
				      <td><%= book.getYear() %></td>
				  </tr>

	<%
			}//for loop
		}// if booklist != null
	%>

	</tbody>
</table>

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

Servlet Codes :

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

	response.setContentType("text/html");
	PrintWriter out = response.getWriter();
	String str = request.getParameter("searchStr");

	List books = null;
	try{
		BookDBAO db = new BookDBAO();
		books = db.getBooksByTitle(str);
		request.setAttribute("books", books);
		 request.getRequestDispatcher("/searchbooks_MYSQL.jsp").forward(request,
		response);

	} catch (Exception ex) {
		 response.resetBuffer();

	}
}
```

4. An AJAX Enabled Web Application with MYSQL and responding with XML data

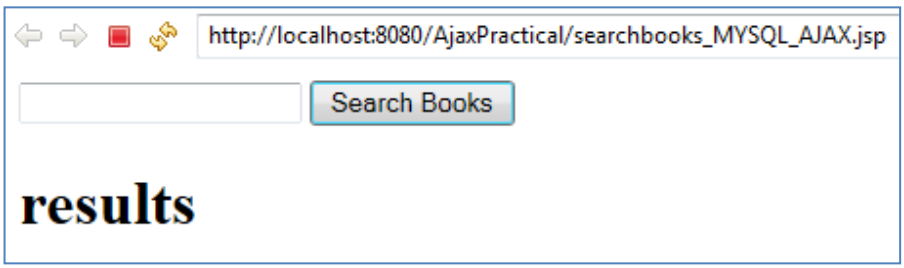
Create a JSP and Servlet that provides a search feature to an online bookstore with MYSQL as datastore integrating AJAX capabilities. The Servlet extract data from Database based on the searchStr and return data as XML format. JSP will reformat the XML data to the requirement HTML table format. Create a Java-based web application with the following files :

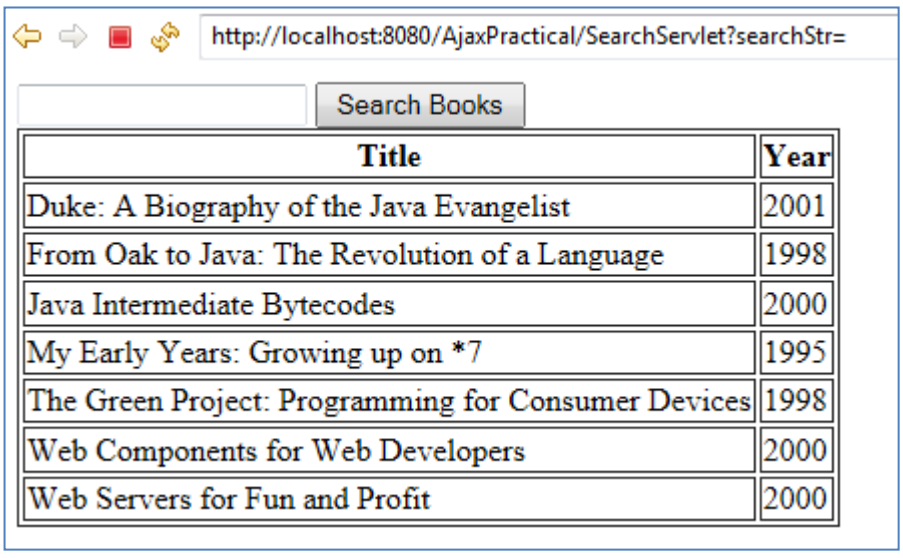
JSP page	**: searchBooks_MYSQL.jsp** (inside **WebContent**)
Servlet	**: SearchServlet.java** (inside package **servlets**) Response : XML ** use the same SearchServlet or create a new one.
Database	**MYSQL – 'test' database**

Screens

To return all results – leave the search string as empty

A total 7 books will be returned.





Title	Year
Duke: A Biography of the Java Evangelist	2001
From Oak to Java: The Revolution of a Language	1998
Java Intermediate Bytecodes	2000
My Early Years: Growing up on *7	1995
The Green Project: Programming for Consumer Devices	1998
Web Components for Web Developers	2000
Web Servers for Fun and Profit	2000

JSP Codes :

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>

<script>

function initRequest() {
       if (window.XMLHttpRequest) {
           return new XMLHttpRequest();
       } else if (window.ActiveXObject) {
           isIE = true;
           return new ActiveXObject("Microsoft.XMLHTTP");
       }
}


function sendRequest() {

       var a = document.getElementById('searchStr');
       var url = "SearchServlet?searchStr=" + escape(a.value);
       var req = initRequest();
       req.onreadystatechange = function() {
          if (req.readyState == 4) {
              if (req.status == 200) {
              document.getElementById("myDiv").innerHTML =  processXML(req.responseXML);
              } else if (req.status == 204){
                        //clearTable();
           }
       }};
       req.open("GET", url, true);
      req.send(null);
}
```

XML Data : (2 books)

```
<Books>
      <Book>
          <Title> Duke: A Biography of the Java Evangelist</Book>
          <Year>2001</Year>
      </Book>
      <Book>
          <Title> From Oak to Java: The Revolution of a Language</Book>
          <Year>1998</Year>
      </Book>

</Books>
```

```
function processXML(responseXML) {

  var books = responseXML.getElementsByTagName("Books")[0];
  var numBooks = books.childNodes.length;
  var myTable = '';

  if (numBooks > 0) {

      myTable += '<Table border="1">';
      myTable += '<thead style="background-color:lightblue">';
      myTable += '<tr><th>Title</th><th>Year</th></tr></thead>';

      myTable += '<tbody>';
      for (loop=0; loop< books.childNodes.length; loop++){
            var aBook = books.childNodes[loop];
            var bookTitle = aBook.getElementsByTagName("Title")[0];
            var bookYear = aBook.getElementsByTagName("Year")[0];

            myTable += '<tr>';
            myTable += '<td>' + bookTitle.childNodes[0].nodeValue + '</td>';
            myTable += '<td>' + bookYear.childNodes[0].nodeValue + '</td>';
            myTable += '</tr>';
      }
            myTable += '</tbody>';
            myTable += '</Table>';

            //alert(myTable);
            return myTable;
  }
}

</script>

</head>
<body>

<input type="text" size="20" id="searchStr" name="searchStr">
<input id="submit_btn" type="Submit" value="Search Books" onClick="sendRequest();">

<div id="myDiv">
<h1>results</h1>
</div>

</body>
</html>
```

Servlet Codes :

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

	response.setContentType("text/xml");
	response.setHeader("Cache-Control", "no-cache");
	PrintWriter out = response.getWriter();
	String str = request.getParameter("searchStr");

	List books = null;
	try{
		BookDBAO db = new BookDBAO();
		books = db.getBooksByTitle(str);

		if(books != null){
			StringBuilder sbXML = new StringBuilder();

			for (int i=0; i<books.size(); i++) {
				BookDetails book = (BookDetails) books.get(i);
				sbXML.append("<Book>");
				sbXML.append("<Title>" + book.getTitle() + "</Title>");
				sbXML.append("<Year>"  + book.getYear() + "</Year>");
				sbXML.append("</Book>");
			}

			out.write("<Books>" + sbXML.toString() + "</Books>");

		}
	} catch (Exception e1){

		response.setStatus(HttpServletResponse.SC_NO_CONTENT);
		response.resetBuffer();
	}
}
```

5. An AJAX Enabled Web Application with MYSQL (AutoComplete Search String)

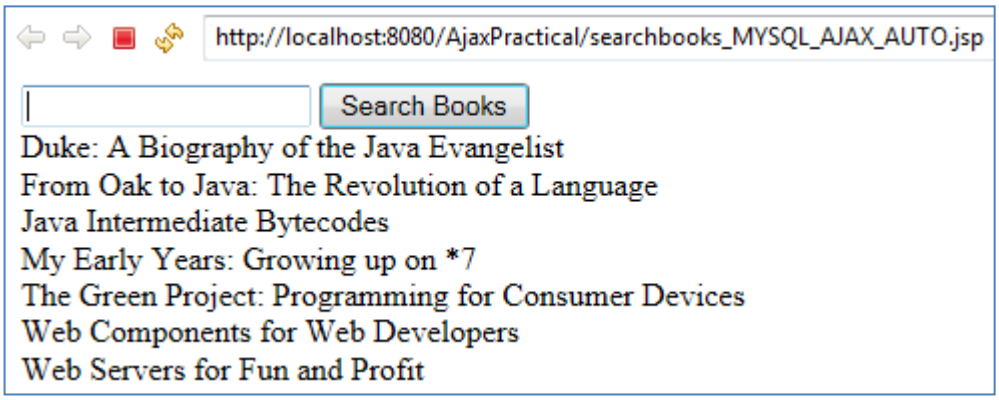
Create a JSP and Servlet that provides an AutoComplete search suggestion feature to an online bookstore with MYSQL as datastore integrating AJAX capabilities. The Servlet extract data from Database based on the searchStr and return data as Text format. A list of suggested Titles will appear below the search bar. Create a Java-based web application with the following files :

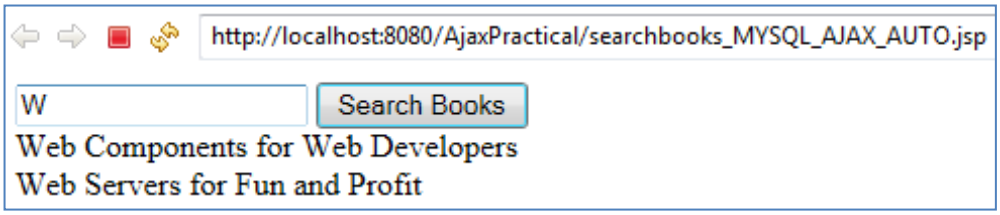
JSP page	**: searchBooks_MYSQL.jsp** (inside **WebContent**)
Servlet	**: SearchServlet.java** (inside package **servlets**) Response : TEXT/HTML ** use the same SearchServlet or create a new one.
Database	**MYSQL – 'test' database**

Screens

To return all suggested titles – leave a space

A total 7 books will be returned.





JSP Codes :

```
<html>
<head>
<title>Insert title here</title>

<script>

function initRequest() {
       if (window.XMLHttpRequest) {
           return new XMLHttpRequest();
       } else if (window.ActiveXObject) {
           isIE = true;
           return new ActiveXObject("Microsoft.XMLHTTP");
       }
   }

function sendRequest() {

       var a = document.getElementById('searchStr');
       var url = "SearchServlet?searchStr=" + escape(a.value);
       var req = initRequest();
       req.onreadystatechange = function() {
          if (req.readyState == 4) {
              if (req.status == 200) {
                    document.getElementById("myDiv").innerHTML =
processXML(req.responseXML);
              } else if (req.status == 204){
                       //clearTable();
              }
       }};
       req.open("GET", url, true);
      req.send(null);
}

function sendRequestAuto(a) {

           var url = "AutoComplete?searchStr=" + a;
           var req = initRequest();
           req.onreadystatechange = function() {
               if (req.readyState == 4) {
                   if (req.status == 200) {
                    document.getElementById("myDivAuto").innerHTML = req.responseText;
                   } else if (req.status == 204){
                       //clearTable();
                   }
               }
           };
           req.open("GET", url, true);
           req.send(null);
}
```

```
function processXML(responseXML) {

  var books = responseXML.getElementsByTagName("Books")[0];
  var numBooks = books.childNodes.length;
  var myTable = '';

  if (numBooks > 0) {

      myTable += '<Table border="1">';
      myTable += '<thead style="background-color:lightblue">';
      myTable += '<tr><th>Title</th><th>Year</th></tr></thead>';

      myTable += '<tbody>';
      for (loop=0; loop< books.childNodes.length; loop++){
            var aBook = books.childNodes[loop];
            var bookTitle = aBook.getElementsByTagName("Title")[0];
            var bookYear = aBook.getElementsByTagName("Year")[0];

            myTable += '<tr>';
            myTable += '<td>' + bookTitle.childNodes[0].nodeValue + '</td>';
            myTable += '<td>' + bookYear.childNodes[0].nodeValue + '</td>';
            myTable += '</tr>';
      }
            myTable += '</tbody>';
            myTable += '</Table>';

            //alert(myTable);
            return myTable;
  }
}


</script>

</head>

<body>

<input type="text" size="20" id="searchStr" name="searchStr"
onkeyup="sendRequestAuto(this.value);">

<input id="submit_btn" type="Submit" value="Search Books"
onClick="sendRequest();">
<div id="myDivAuto">
</div>

<hr/>

<div id="myDiv">
<h1>results</h1>
</div>
```

Servlet Codes :

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

	response.setContentType("text/xml");
	response.setHeader("Cache-Control", "no-cache");
	PrintWriter out = response.getWriter();
	String str = request.getParameter("searchStr");

	List books = null;
	try{
		BookDBAO db = new BookDBAO();
		books = db.getBooksByTitle(str);

		if(books != null){
			StringBuilder sbXML = new StringBuilder();

			for (int i=0; i<books.size(); i++) {
				BookDetails book = (BookDetails) books.get(i);
				sbXML.append("<Book>");
				sbXML.append("<Title>" + book.getTitle() + "</Title>");
				sbXML.append("<Year>"  + book.getYear() + "</Year>");
				sbXML.append("</Book>");
			}

			out.write("<Books>" + sbXML.toString() + "</Books>");

		}
	} catch (Exception e1){

		response.setStatus(HttpServletResponse.SC_NO_CONTENT);
		response.resetBuffer();
	}
}
```

```
	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		// TODO Auto-generated method stub


		response.setContentType("text/html");
		PrintWriter out = response.getWriter();
		String str = request.getParameter("searchStr");

		List books = null;
	    try{
			BookDBAO db = new BookDBAO();
			books = db.getBooksByTitle(str);

			if(books != null){
```

```
                        String buffer = "<div>";

                        for (int i=0; i<books.size(); i++) {
                              BookDetails book = (BookDetails)books.get(i);
                              buffer += book.getTitle() + "<br>";

                        }
                        buffer += "</div>";
                        out.println(buffer);

                  }
            } catch (Exception e1){

                  response.setStatus(HttpServletResponse.SC_NO_CONTENT);
                  response.resetBuffer();
            }

      }
```

Conclusion

What is particularly attractive about this is that AJAX applications do not require a separate plug-in, and are platform and browser-neutral. That said, AJAX is not supported as well in older browsers. Care needs to be taken in writing client-side script that accounts for the differences between browsers. You might consider using a JavaScript library that abstracts the browser differences and in some cases support older browsers using alternative interaction techniques.