

## **Homework: AllMusic& Facebook Mashup – an AJAX/JSON/Java Exercise**

### **1. Objectives**

- Become familiar with the AJAX, JSON& XML technologies.
- Use a combination of HTML, CSS, DOM, XMLHttpRequest, XML and Java Servlets.
- Provide an interface to perform discography search from allmusic.com and post details to Facebook.

### **2. Background**

#### **2.1 AJAX & JSON**

AJAX (**A**synchronous **J**avaScript + **X**ML) incorporates several technologies:

- Standards-based presentation using XHTML and CSS;
- Dynamic display and interaction using the Document Object Model (DOM);
- Data interchange and manipulation using XML and XSLT;
- Asynchronous data retrieval using XMLHttpRequest;
- JavaScript binding everything together.

See the class slides at <http://www-scf.usc.edu/~csci571/Slides/ajax.ppt>.

JSON, short for JavaScript Object Notation, is a lightweight data interchange format. Its main application is in AJAX web application programming, where it serves as an alternative to the use of the XML format for data exchange between client and server. See the class slides at <http://www-scf.usc.edu/~csci571/Slides/JSON1.ppt>.

#### **2.2. AllMusic**

<http://www.allmusic.com/> is an online database for music albums, artists and songs; including reviews and biographies.

In Homework #6 a Perl or PHP script together with your Apache server provided the “scrape” functionality. In this exercise you will re-use your scraping code to produce XML instead of HTML as you did in homework #6.

#### **2.3 Facebook**

**Facebook** is a global social networking website that is operated and privately owned by

Facebook, Inc. Users can add friends and send them messages, and update their personal profiles to notify friends about themselves and what they are doing.

Users can additionally post news feeds to their profiles, and these feeds may include images, besides text messages.

The Facebook homepage is available at: <http://www.facebook.com>

Facebook provides developers with an API called the **Facebook Platform**. **Facebook Connect** is the next iteration of Platform, which provides a set of API's that enable Facebook members to log onto third-party websites, applications and mobile devices with their Facebook identity. While logged in, users can connect with friends via these media and post information and updates to their Facebook profile.

Below are few links for Facebook Connect:

<http://developers.facebook.com/blog/post/108/>

<http://developers.facebook.com/docs/guides/web/>

### 3. Description of the Exercise

In this exercise, you will write a web application that does the following sequence of actions:

- Allow a user to enter a “query” regarding information from allmusic.com; the query will contain a title and a type. The type can be one of the following: Artist, Album, Song;
- Use the query string to retrieve appropriate information from allmusic.com, using the modified Perl or PHP script from Homework #6;
- Display the discography details in a table format similar to the format used in Homework #6. (Each entry in the table should now be accompanied by a Facebook button);
- Allow the user to select a particular entry and Post it to Facebook by clicking on the “Facebook” button;
- Authorize the user to login to Facebook;
- Post a feed of the artist/album/song information to the user’s Facebook wall using the Facebook Connect API.

The format of the feed to be posted is as follows:

For Artists:

**“artist\_name**

I like **artist\_name** who is active since **year**.

Genre of Music is: **genre**.

Look at details: **here**”

Enter appropriate values for **artist\_name**, **year**, **genre**. The **artist\_name** (first line) and **here** should hyperlink to the Artist details page on allmusic.com.

For Albums:

**“album\_name**

I like **album\_name** released in **year**.

Artist: **artist** Genre : **genre**

Look at details: **here**”

Enter appropriate values for **album\_name**, **year**, **artist**, **genre**. The **album\_name** (first line) and **here** should hyperlink to the Album details page on allmusic.com.

For Songs:

**“song\_title**

I like **song\_title** composed by **composer**

Performer: **performer**

Look at details: **here**”

Enter appropriate values for **song\_title**, **composer**, **performer**. The **song\_title** (first line) and **here** should hyperlink to the song details page on allmusic.com.

In all of the above case, users can enter the message of their choice in the message box.

(Refer to Figure 3 for more details on the format of the message to be displayed)

A snapshot of the initial user interface, together with a query and the resulting table, is shown in Figure 1.

## Discography Search

Title :

Type :

Search

## Displaying 5 results for TAYLOR





Cover	Name	Genre(s)	Year	Details	Post To Facebook
	Taylor Swift	Country, Pop/Rock	2000s - 2010s	<a href="#">details</a>	
	James Taylor	Pop/Rock	1960s - 2010s	<a href="#">details</a>	

Figure 1 – User Interface

To implement this exercise you are required to write a combination of HTML, JavaScript and Java Servlet programs. The top-level interface consists of three areas:

- A form including a text area to enter the title, a dropdown to select the type of information you want to search and a “Search” button;
- A dynamic area that displays a table with the appropriate information corresponding to the query (Table from Homework 6);
- A “Facebook” button that posts the details, corresponding to the selected entry, to a user’s feed page.

Once title has been entered in the edit box along with the selection of the type, and the “Search” button is pressed, the form calls a JavaScript function. This function first performs basic validation of the input. If the input is empty, the request is not forwarded to the Java Servlet.

Instead an alert with an error message is presented to the user, asking to refine the search. Once the validation is successful, the JavaScript function executes XMLHttpRequest to start an asynchronous transaction with a Java Servlet running under Tomcat, and passing the “query strings” as parameters of the transaction.

The Java Servlet in turn performs the following three steps:

First, the Java Servlet extracts the query string and then it calls the Perl or PHP script, modified after Homework #6, to retrieve data from allmusic.com corresponding to the title and type included in the query. For example, if your server was using port XXXX, a query of the following type will be generated:

If title = adele, type = artists

<http://cs-server.usc.edu:XXXX/cgi-bin/script.pl?title=adele&type=artists>

Secondly, the Perl or PHP script would return the XML of the following type:

```
<results>
<result cover="http://cps-
static.rovicorp.com/3/JPG_170/MI0003/092/MI0003092113.jpg?partner=allrovi.com" name="Adele" genre=" P
op/Rock, R&B" year="2000s - 2010s" details="http://www.allmusic.com/artist/adele-mn0000503460"/>
<result cover="http://cps-
static.rovicorp.com/3/JPG_170/MI0003/208/MI0003208257.jpg?partner=allrovi.com" name="Adele
Anthony" genre="Classical" year="1990s - 2010s" details="http://www.allmusic.com/artist/adele-anthony-
mn0000059205"/>
<result cover="NA" name="Adele &
Glenn" genre="Pop/Rock " year="2010s " details="http://www.allmusic.com/artist/adele-amp-glenn-
mn0002943666"/>
<result cover="NA" name="Adele
Kozak" genre="Classical" year=" 2010s " details="http://www.allmusic.com/artist/adele-kozak-
mn0002794488"/>
<result cover="NA" name="Adele Schmidt" genre=" Stage &
Screen" year=" 2010s" details="http://www.allmusic.com/artist/adele-schmidt-mn0002977946"/>
</results>
```

Notice that in Homework #6 your Perl or PHP script produced HTML. In this exercise, the output must be changed to XML.

Now, the Java Servlet extracts the **appropriate information** from this XML.

Lastly, the Java Servlet produces a JSON string that is returned asynchronously to the original XMLHttpRequest.

The format of the JSON string that needs to be generated is as follows:

```
{
  "results":{
    "result":{
```

```

        "cover":"http://cps-
static.rovicorp.com/3/JPG_170/MI0003/092/MI0003092113.jpg?partner=allrovi.com",
"name":"Adele", "genre":"Pop/Rock,R&B", "year":"2000s-2010s",
"details": http://www.allmusic.com/artist/adele-mn0000503460
    },
    {
        "cover":"http://cps-
static.rovicorp.com/3/JPG_170/MI0003/208/MI0003208257.jpg?partner=allrovi.com",
"name":"Adele Anthony", "genre":"Classical", "year":"1990s-2010s",
"details":http://www.allmusic.com/artist/adele-anthony-mn0000059205
    },
    {
        "cover":"NA", "name":"Adele & Glenn", "genre":"Pop/Rock",
"year":"2010s", "details":"http://www.allmusic.com/artist/adele-amp-glenn-mn0002943666"
    },
    {
        "cover":"NA", "name":"Adele Kozak", "genre":"Classical",
"year":"2010s", "details":http://www.allmusic.com/artist/adele-kozak-mn0002794488
    },
    {
        "cover":"NA", "name":"Adele Schmidt", "genre":"Stage & Screen",
"year":"2010s", "details":"http://www.allmusic.com/artist/adele-schmidt-mn0002977946"
    }
}
}

```

After obtaining the query results from the callback of XMLHttpRequest, the JavaScript program displays the appropriate table (as per Homework #6) in the “dynamic” area of the web page. **Also note, successive queries will clear the data of the dynamic area and overwrite it with new data.**

Next, the user is allowed to click on the “Facebook” button adjacent to a particular entry in the table. When the button is pressed, the web application does the following:

- Authorizes the user to Facebook (i.e. logs him/her in) using the application and user credentials if the user is not already logged in to Facebook;
- Posts an Update Status message to the feed (described at the beginning of section 3).
- The above two steps can be performed using the Facebook Connect API, using the JavaScript SDK, which provides a rich set of client-side functionality for accessing Facebook's server-side API calls.

It is documented at: <https://developers.facebook.com/docs/reference/javascript/>

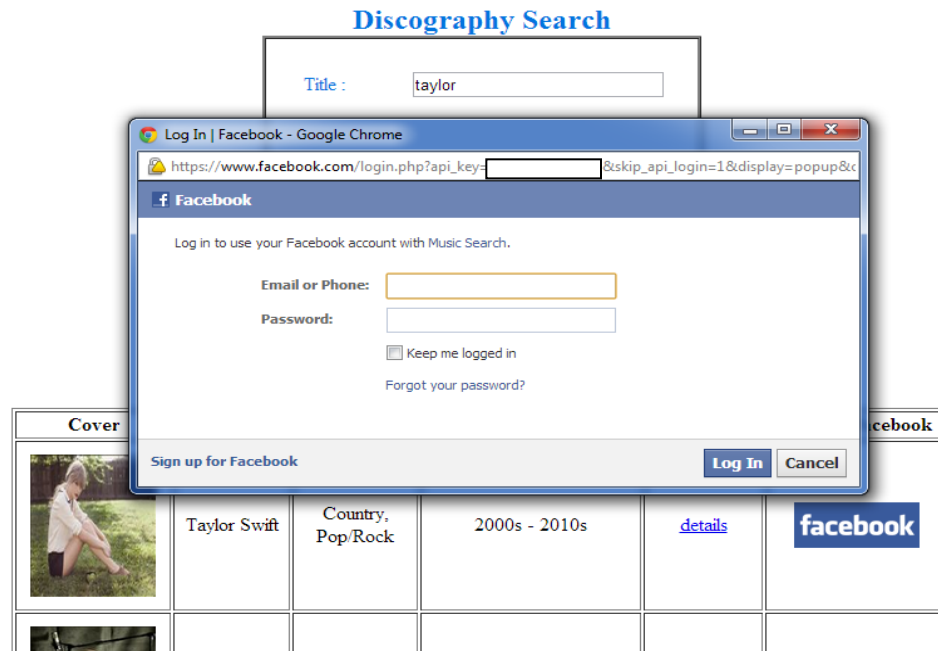


Figure 2 – Facebook Login

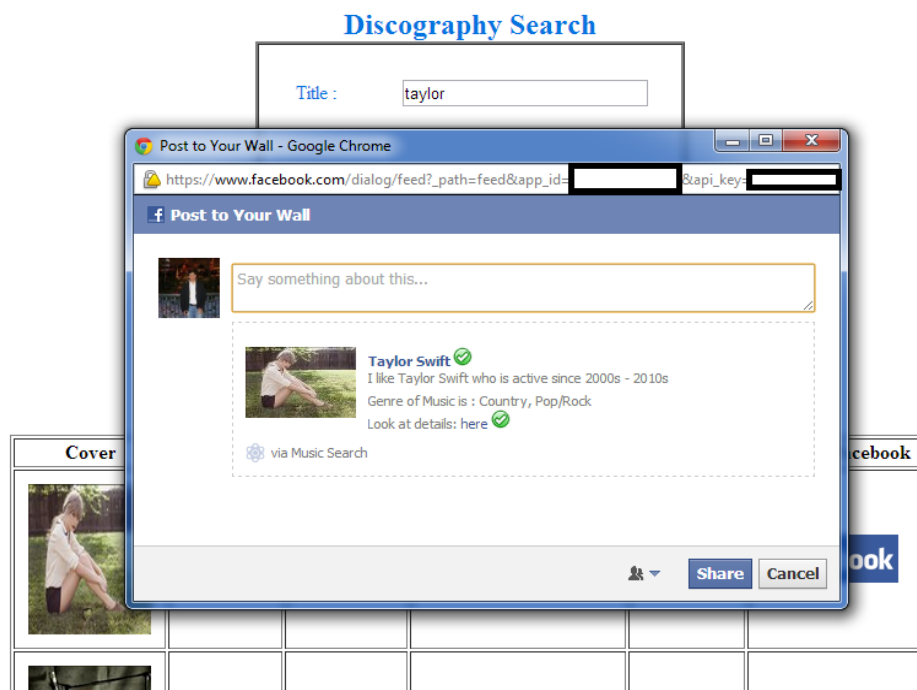


Figure 3 – Post to user's feed

## 4. Implementation Hints

- *Step 1: Writing your JavaScript Program – set up an Ajax transaction*

The JavaScript invoked by the Search button click event should do all of the following:

- Assign the “callback” function;
- Assemble the “url” parameter of the GET request as a reference to the Java Servlet to be invoked:

Example url : <http://cs-server.usc.edu:XXXXX/servlet/MyServlet?parameters>

- Call the XMLHttpRequest method (see Ajax Slide 24) and create the request object.
- Prepare the GET XMLHttpRequest using the setRequestHeader method:

```
req.open("GET", url, true);  
  
req.onreadystatechange = myCallback;  
  
req.setRequestHeader("Connection", "Close");  
  
req.setRequestHeader("Method", "GET" + url + "HTTP/1.1");
```

- *Step 2: Writing your JavaScript Program – Execute Ajax Transaction*

The JavaScript should finally invoke the XMLHttpRequest `send` method (see Ajax slide 24).

The “callback” function should check for completion of the transaction (request `readyState` equal to 4 and `status` equal to 200 (see AJAX slide 27 and JSON slide 5); use `eval()` and the `responseText` to retrieve the resulting JSON data (see JSON slide 5), and display the returned information properties to the “dynamic” area.

- *Step 3: Use the Java Servlet to respond to XMLHttpRequest and retrieve the information listings from allmusic.com*

The Java Servlet referred above (in step 1) as `/servlet/MyServlet` (see 1.c above) should be invoked using `doGet()`.

The Java Servlet should do all of the following:

- Initiating a connection with the Perl or PHP script, using the Apache server from Homework #6, to retrieve the appropriate listings by scraping allmusic.com.



- *Step 4: Use the Java Servlet to retrieve the XML file content*

You may have to use an XML parser (for example: JAXP). Steps to retrieve XML file contents are as follows:

Step a: Get the XML content based on the URL above in Step 3.a.

- You need to open a URL connection to get the file you want.

To create a URL connection:

```
URL url = new URL(urlString);

URLConnection urlConnection = url.openConnection();

urlConnection.setAllowUserInteraction(false);

InputStream urlStream = url.openStream();

//read content
```

Step b: Parse the XML file using an XML parser

- Any XML parser can be used to parse the XML file. You can use methods like `getNodeName()` to access these elements. A good choice might be the JDOM library, which you get from: <http://www.idom.org/downloads/index.html>

- *Step 5: Use the Java Servlet to process the XML data*

As you parse the data, you will build an output string, converting the XML data into JSON format, as described in section 3.

Finally you will return the JSON as a single string to the calling JavaScript program. To easily create a JSON string, you might find useful the JSON-RPC library available at:

<http://mirrors.ibiblio.org/pub/mirrors/maven/com.metaparadigm/jars/json-rpc-1.0.jar>

The Java Servlet should handle exceptions such as `MalformedURLException` and `IOException`. The Java Servlet should also handle error responses sent from the Apache servlet and reply with an appropriate error, a JSON message to the initial JavaScript `XMLHttpRequest`. This way, the JavaScript callback function will be able to inform the user that an error has occurred.

- *Step 6: Writing your JavaScript Program – Post media details to Facebook*

Once the media properties are displayed in the dynamic area, the user should be able to click on the “Facebook” button for the corresponding entry in the table.

- *Step 7: Writing your JavaScript Program – Authorize Facebook User*

Once the user clicks on the Facebook button, the program invokes the Facebook Connect API and authorizes the user. The recommended API to use is FB.Init, which is documented at: <http://developers.facebook.com/docs/reference/javascript/FB.init/>

Also look at the code listed under “Loading” in the JavaScript SDK page at:

<https://developers.facebook.com/docs/reference/javascript/>

- *Step 8: Writing your JavaScript Program – Post media information to Facebook News Feed*

There are several methods to post a message to the user’s feed page (the “wall”).

One such method uses the Fb.ui() API with the feed dialog, documented at:

<http://developers.facebook.com/docs/reference/javascript/FB.ui/>

The feed dialog is documented at:

<https://developers.facebook.com/docs/reference/dialogs/feed/>

Once the user is authorized, and an appropriate session token has been obtained, the text of the selected entry is posted to the user’s news feed page (the “wall”). A subsequent posting will not require the user to log in again.

Additional information that may be useful to you is available at “Facebook for Websites” web page: <https://developers.facebook.com/docs/guides/web/>

## 5. Prerequisites

This homework requires the use of the following components:

- A servlet-based web server, Tomcat 4.1.27. Instructions on how to load Tomcat 4.1.27 can be found here:  
<http://www-scf.usc.edu/~csci571/2006Spring/tomcatinstall.html>.  
A tar version of Tomcat 4.1.27 can be found here:  
<http://www-scf.usc.edu/~csci571/download/jakarta-tomcat-4.1.27.tar>.
- The Java Servlet library (servlet-api.jar) to perform HTTP transactions using methods such as doGet() or doPost() from Java.  
You may find it here :<http://repo1.maven.org/maven2/javax/servlet/servlet-api/2.5/>
- A Java XML parser library. You may use the JDOM, an object model that uses XML parsers to build documents, available in the Download section of the jdom website  
<http://www.jdom.org/downloads/index.html>
- You may also use JAXP, the Java API for XML Parsing. A good tutorial on JAXP is available at <http://www-106.ibm.com/developerworks/xml/library/x-jaxp1.html>

or <http://docs.oracle.com/javase/tutorial/jaxp/intro/index.html>

- You need to create a Facebook Platform application:

To do that you will need to add the Facebook Developer application: go to <https://developers.facebook.com/> and in the apps section, click **Create New App**. Once you've added the Facebook Developer application to your account, you can begin creating your application for Facebook. You should be getting an **API Key** and **Application Secret** that you will have to use with the JavaScript Client Library FB.init API.

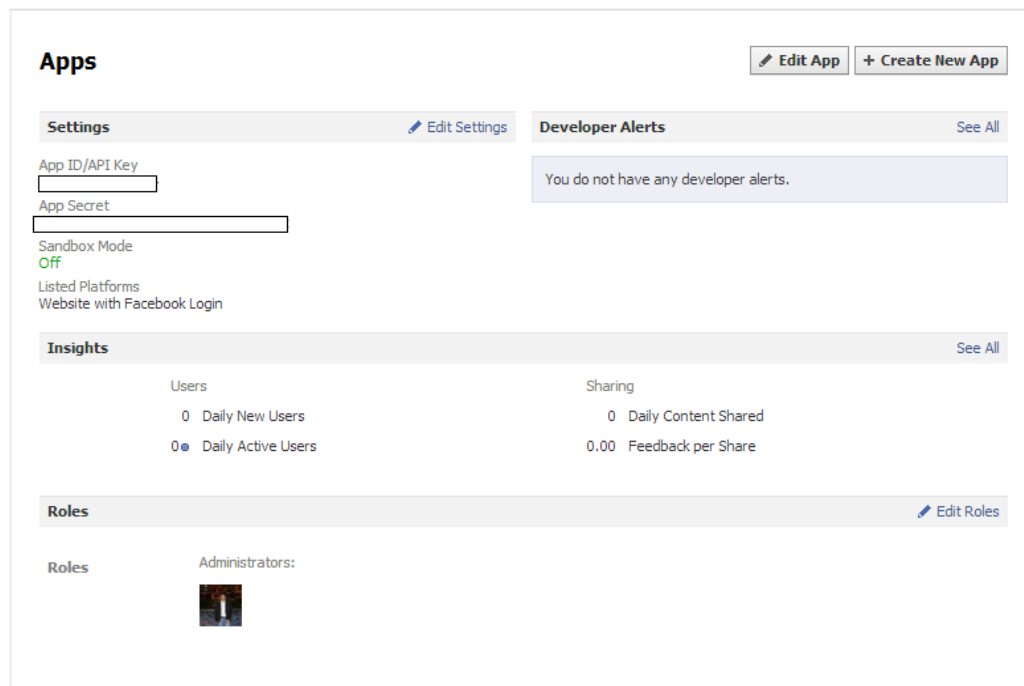


Figure 4 – Sample Facebook Developers App Page

## 6. Deployment Structure

To write your own Java Servlets program using Tomcat 4.1.27, you need to:

- Successfully install Tomcat 4.1.27 on your machine.
- Go to \$CATALINA\_HOME/webapps/examples directory.
- Place the HTML, CSS and JavaScript (.js) files in the Tomcat servlets subdirectory.
- Place your Java Servlets file (.java) in the /WEB-INF/classes folder. So the path of your Servlets file is [http://server\\_name:port/examples/servlet/your\\_servlet\\_name](http://server_name:port/examples/servlet/your_servlet_name)
- Add appropriate sections to the WEB-INF/web.xml file, as in:

```
<servlet>
<servlet-name>music_search</servlet-name>
<servlet-class>MusicSearch</servlet-class>
</servlet>
```

```
<servlet-mapping>
<servlet-name>music_search</servlet-name>
<url-pattern>/servlet/musicsearch</url-pattern>
</servlet-mapping>
```

- To avoid UTFDataFormatException during file IO operation, you have to use JDK 1.3 or later for Tomcat. In the .cshrc file under your home directory, add the entries:

```
setenv JAVA_HOME /usr/j2se
setenv PATH /usr/j2se/bin:${PATH}
```

- Before you issue a request to your Java Servlet file, you need to compile it. You will need a Java Servlet class to compile your code, so open the .cshrcfile, and add the full path to the Tomcat file that implements the Servlet class APIs located in ../jakarta-tomcat-4.1.27/common/lib/servlet.jar to your CLASSPATH variable, and use the variable with the classpathswitch of the javac compiler.
- Then run “source .cshrc” and you are ready to compile your Java files.

## 7. Material You Need to Submit

On your course homework page, your link for this homework should go to a page that includes your JavaScript/HTML program. You should submit all source code files including HTML (.html), Cascading Style Sheets (.CSS), JavaScript (.js), perl or PHP scripts(.pl or .php) and Java Servlets (.java) electronically to the csci571 account so that it can be graded and compared to all other students' code via the MOSS code comparison tool.

## 8. Partial Credit

If you complete Step 1 through 6 as listed in Section 4, you will obtain 5 points. If you also complete Steps 7 and 8 (authorizing a user to Facebook and posting the message to the user's Facebook news feed), you will obtain the full 10 points for the exercise.