

Let's chat with Ajax

Skill Level: Intermediate

[Judith M. Myerson \(jmyerson@bellatlantic.net\)](mailto:jmyerson@bellatlantic.net)
System Engineer and Architect

17 Mar 2009

Want to chat with Asynchronous JavaScript and XML (Ajax)? Wish you could have a dedicated, open source Web chat pop up in response to a system event and let you know what's happening—for example, when performance goes below the guaranteed service level? Regular developerWorks author Judith Myerson introduces the idea of a two-panel chat for systems administrators to exchange private messages on one side and broadcast messages to general users on the other side. She offers solutions for chat server overload and talks about the issues of downloading Ajax Chat, how to change configurations, and even how to add as many channels as you want.

Introduction

Some of you who have used instant messaging chat software have been looking for free, open source Web chat software where you could use your own code to customize the look and feel of the chat and also how it sends messages to the databases on the server. Your search is over with Ajax Chat, an application that's implemented in JavaScript, PHP and MySQL, and is released under the GNU Affero General Public License.

Ajax Chat comes in two versions—stand-alone or forum integration. You can add or limit channels as much as you like. You are not restricted to the maximum number of chat channels you can have in a hosted environment, and there is no cost for adding channels (as well as users and databases) as long as existing resources are not impacted.

With Ajax Chat, you can send private or broadcast messages, delete messages inside the chat, define open hours for the chat, set a length counter, and customize the layout using Cascading Style Sheets (CSS) and a template system. Ajax Chat

has security in mind to prevent code injections, SQL injections, cross-site scripting, session stealing, and other attacks. Just make sure you and your system can provide countermeasures for any chat vulnerabilities hackers could exploit.

Channel wishlist

You could use one of the channels as a dedicated chat for system and user administrators to send and receive real-time alerts on the system status. For example, you can have this channel triggered by a system event to pop up, then sound an alarm and shout at you to get your immediate attention if it thinks you are away from the client for several seconds. Some examples of system events include:

- When the performance begins to fall below a guaranteed service level specified in a Service Level Agreement (SLA)
- Suggested tools to bring the performance back to the service level
- Status of recovery in progress
- When the performance is back to the guaranteed service level
- How long the performance was below the service level and where in the system it happened

The wishlist includes a second panel in the administrator's chat to broadcast messages to users on, for instance, the status of performance level or when socket servers need to be shut down for maintenance or repair and when they would run again. When the system is running normally, the developers and administrators could use the second panel to point to industry-wide open source Ajax libraries.

You must ensure that chat clients can establish a permanent connection to the socket server to listen for chat messages. Because, by default, only local clients (127.0.0.1,::1) may broadcast messages, you may need to change the defaults to allow other clients (for example, your system administration colleagues) to send broadcast messages.

Download Ajax Chat

To start, go to Blueimp's [Ajax Chat](#) at SourceForge.net. You will be shown a package of several files. The first file you should download and unzip is, of course, the main chat application file: `ajax_chat-0.8.1.2.tar.gz`.

Extracting the unzipped files will automatically place the files in the following folders: `css`, `flash`, `img`, `js`, `lib`, `socket`, `sounds`, and `src`. The other files to be downloaded will integrate Ajax Chat with `phpBB2`, `phpBB3`, `MyBB`, `PunBB`, `SMF`, `vBulletin`, and other

PHP community files.

Before you upload and install the server-side chat files, you need to edit three configuration settings: database, channel, and user. After you upload the files, create database tables, and then delete the installation script. You must install MySQL on the server.

Database settings

You can set up Ajax Chat as a stand-alone version on your server or use one of the forum integration versions. But, you need to change configuration defaults to strengthen your security posture.

Using the stand-alone version

To use Ajax Chat, first go to the `lib/config.php` file. If you want to use Ajax Chat as a stand-alone version on your server, set the host, user, pass, and name of the database to your localhost, user name, password, and database name. Listing 1 shows how it's done.

Listing 1. Database settings for the stand-alone version

```
$config['dbConnection']['host'] = 'localhost';  
$config['dbConnection']['user'] = 'your_database_username';  
$config['dbConnection']['pass'] = 'your_database_password';  
$config['dbConnection']['name'] = 'your_database_name';
```

By default, the database settings `type` and `link` are set to null. If `type` is set to null, it defaults to *mysqli*, or you can change it to *mysql*. You can set `link` as a reference to an existing database connection link or object. If you set `link` to null, a new database connection is created.

Using a forum integration version

If you'd rather use one of the forum integration versions, you can get Ajax Chat to use the existing database connection of your forum by setting all database configuration settings to null. If you are using a different database for the forum integration version, you must comment out a line in the `lib/class/CustAJAXChat.php` file. For instance, for the phpBB3 integration version, you will have to comment out the following:

```
$this->setConfig('dbConnection', 'link', $db->db_connect_id);
```

When you are done, you can set the database settings like you did with the stand-alone version.

If you want to use the chat as a shoutbox in your forum, you need to change the URL and path of the chat directories. To do this, you first go to the `includes/functions.php` file and look for the URL and path to the chat folder, such as:

```
define('AJAX_CHAT_URL', '/chat/');  
define('AJAX_CHAT_PATH', realpath(dirname($_SERVER['SCRIPT_FILENAME']).'/chat').'/');
```

In both instances, fix `'/chat/'` to read `'../chat/'`.

Channel settings

In the `lib` directory, navigate to the `data` subdirectory and look for the `channels.php` file. Suppose you want three channels, each with a unique ID and name. Listing 2 shows how this would look.

Listing 2. Channel settings

```
$channels=array()  
$channels[0]='Public'  
$channels[1]='123'  
$channels[2]='456'
```

Suppose then that you expand to 20 channels. To save time in editing the `channels.php` file, you can limit them to three channels in the `config.php` file, like this:

```
$config['limitChannelList'] = array(123,456,789);
```

This setting says that only IDs 123, 456, and 789 can be used as channels, skipping the other 17 channels. To add these channels as custom chat channels, you need to adjust the methods `getChannels()` and `getAllChannels()` in the `lib/class/CustomAJAXChat.php` file. The method `getChannels()` returns an array of channels to which the current user has access. The method `getAllChannels()` returns an array of all available channels whether or not the current user has access to them.

To do this, you'd first add the following code to the `getChannels()` method:

```
$this->_channels = array_merge($this->_channels,  
array('Extra_Public_Channel_1'=>123, 'Extra_Public_Channel_2'=>456,  
      'Extra_Public_Channel_3'=>789));
```

before

```
} return $this->_channels;
```

Do the same for the `getAllChannels()` method by replacing `_channels` with `_allChannels`.

Edit settings

In the data subdirectory, look for the `user.php` file. Make sure each user has a unique ID and a unique name. The first user in the list is set as the guest user, and all guest users have access to the channels set for this user and user role `AJAX_CHAT_GUEST`. They are not assigned a user name and password.

Listing 3. Default guest user

```
// List containing the registered chat users:
$users = array();

// Default guest user
$users[0] = array();
$users[0]['userRole'] = AJAX_CHAT_GUEST;
$users[0]['userName'] = null;
$users[0]['password'] = null;
$users[0]['channels'] = array(0);
```

Do not delete the user settings for the guest user when you grant user roles for the registered users, moderators, and administrators. Listing 4 shows sample user settings for an administrator.

Listing 4. Admin user

```
$users[1] = array();
$users[1]['userRole'] = AJAX_CHAT_ADMIN;
$users[1]['userName'] = 'admin';
$users[1]['password'] = 'admin';
$users[1]['channels'] = array(0,1);
```

Make sure you change the user name default and give it a strong password.

To grant users admin/moderator rights, adjust the `getValidLoginUserData()` method in the `lib/class/CustomAjaxChat.php` file. For example, to grant moderator rights to a specific user with the user ID 123456, you could add

```
if($userData['userID'] == 123456)
    $userData['userRole'] = AJAX_CHAT_MODERATOR;
```

before

```
return $userData;
```

Go to the config.php file and add:

```
$config['customModeratorList'] = array(123,456,789);
```

before

```
?>
```

Then add the following code to the `getValidLoginUserData()` function in the `lib/class/CustomAJAXChat.php` file:

```
if($this->getConfig('customModeratorList') && in_array($userData['userID'],  
    $this->getConfig('customModeratorList'))) {  
    $userData['userRole'] = AJAX_CHAT_MODERATOR;  
}
```

before

```
return $userData;
```

Upload files

Upload the files to your server under your document root if you are using the forum integration version; for example, `http://example.org/forum/chat`.

Suppose you want to use the chat in a different directory than inside the forum directory like:

```
http://example.org/chat  
http://example.org/forum
```

Go to the `lib/custom.php` file and change `$phpbb_root_path = AJAX_CHAT_PATH.'../';` to `$phpbb_root_path = AJAX_CHAT_PATH.'../forum/';` .

Create database tables

Next, you create database tables in the chat folder. You can either execute

install.php on the server or use phpMyAdmin to create the tables. You can edit chat.sql before you use it with the installation script or phpMyAdmin. When you are done, delete install.php and place a link to the chat folder on your Web site.

Ajax Chat problems and solutions

Ajax Chat is designed with several factors that make it resource-efficient. First, the operations the server has to perform are not complex, and the database queries are simple. Second, most of the work is done on the client side—JavaScript handles the code, executes hyperlinks, and displays custom Emoticons. And, third, only updated data from the server is sent to the clients, to keep the traffic low.

One problem that may arise is that when server load is high, the resource is not as efficient in pushing updated data from the server to the client. Another problem is that the default configurations leave security holes that hackers can exploit.

Solution for server overload

One way of reducing the server load is to use a socket server. This is accomplished by installing and using a Flash- (client-side) and Ruby- (server-side) based socket connection to let chat clients permanently pull updates from the server. Events pushed from the server side need a permanent or long-lasting socket connection between the clients and server. This requires either a custom HTTP server (called "comet") or another custom socket server—not a very efficient way, in my opinion, for the server to push the updated data.

To get around this problem, Ajax Chat uses a JavaScript-to-Flash bridge to establish a permanent socket connection from the client side. You must be able to run a Ruby script as a service to run the socket server. To initially start the service, you need to execute the script files in the socket directory, like this:

```
chmod +x server
chmod +x server.rb
```

where `server` is a bash script and `server.rb` is the ruby socket server script.

To start and stop the service at other times, execute `./server start` and `./server stop`. If stopping the chat server doesn't work, try killing `$PID` (process ID of the service) with the `-HUP` options.

To make sure the server-side chat script is broadcasting chat messages through the socket server, set `$config['socketServerEnabled']` to true in the `lib/config.php` file. If you are broadcasting messages and your socket server is running on another host, you should set the broadcast clients option to the chat

server IP.

Solution for socket-less server

Suppose the socket server is down for maintenance or repair. You want to emulate a socket server you temporarily do not have, and you want to keep traffic from the server low and improve response time from the client side. The time between update calls does not exist.

You need to configure time between update calls, delay the chat update, and set the maximum number of users. First, configure the time for 3 seconds to retrieve new chat messages in the `js/config.js` file, like this:

```
timerRate: 3000
```

The default time is 2 seconds.

Then, delay the chat update in the JavaScript file by setting `StartChatOnLoad` to `false`. The chat update is delayed until the event of starting the chat is executed when the chat is loaded.

You can configure the maximum number of users online in the `lib/config.php` file, like this:

```
$config['maxUsersLoggedIn'] = 80;
```

Setting the maximum number of users in the chat does not affect the maximum number of moderators or administrators. The default maximum is 100.

Solution for Flash security errors

When using the JavaScript-to-Chat bridge, you need permissions for creating sockets using Flash. For Flash versions greater than 9.0.115.0, an explicit permission (using XML-syntax) is required for creating socket connections. If you are using earlier Flash versions, you will get a Flash security error message in the browser, even when you activate the socket server.

A solution is to use a policy-files server that will listen to connections in port 843 in the server. Each time a client tries to connect to the chat, the Flash client will request the policy authorization to the server. If needed, you can download a [Flash security policy](#).

If you are annoyed with security errors you are receiving while trying to fix the problem, you can suppress these error messages by commenting out the following line in the `js/chat.js` file:

```
setTimeout('ajaxChat.addChatBotMessageToChatList(\'/error SocketSecurity\')', 0);
```

Conclusion

This article helps you to add a chat system to your environment. Potential users' demand for an open source chat that would pop up in response to a system event when, for example, the performance goes below the guaranteed service level, presents a challenge for the developers and other members of a project team who look for ways of making their job easier. Being aware of and resolving with the issues of designing Ajax Chat and potential project risks can make your team's experiences trouble-free. You can accomplish this by using IBM Rational® Web Developer WebSphere® Software to build Ajax applications and IBM Rational ClearQuest for defect and application tracking.

Resources

Learn

- To get more help in installing and configuring Ajax Chat go to the Wiki section of [SourceForge.net](#).
- Read more about the [GNU Affero General Public License](#).
- Want to look up industry-wide Ajax libraries? Find them in the [OpenAjax Registry](#).
- The [Work with Web services in enterprise-wide SOA series](#) by Judith M. Myerson offers information on how to work with Web services in enterprise-wide SOAs.
- Browse the Judith M. Myerson's series, [Use SLAs in a Web services context](#), for details on service-level agreements.
- Want more information on Ajax tools? Read about them in "[Survey of Ajax tools and techniques](#)" (developerWorks, July 2007).
- Read "[Tight coupling Web services in the SOA](#)" (developerWorks, Jan 2008) for a look at the pros and cons of both tight and loose coupling Web services.
- Get the business insight and the technical know-how to ensure successful systems integration by reading [Enterprise Systems Integration, Second Edition](#).
- Visit the [technology bookstore](#) for books on these and other technical topics.
- Stay current with [developerWorks technical events and webcasts](#).

Get products and technologies

- See how [IBM Rational Web Developer for WebSphere Software](#) for architecture management, [IBM Rational ClearQuest](#) for change and release management, and [IBM Rational Functional Tester Plus](#) for quality management can help when developing Ajax and other applications. These tools from IBM help increase your productivity by reducing testing time and the costs of test labs in your enterprise
- [IBM trial products for download](#): Build your next development project with IBM trial software, available for download directly from developerWorks.

Discuss

- [developerWorks blogs](#): Get involved in the developerWorks community.

About the author

Judith M. Myerson

Judith M. Myerson is a systems architect and engineer. Her areas of interest include middleware technologies, enterprise-wide systems, database technologies, application development, network management, security, RFID technologies, and project management.

Trademarks

IBM, the IBM logo, ibm.com, DB2, developerWorks, Lotus, Rational, Tivoli, and WebSphere are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. See the current list of IBM trademarks.