

Registration and Log-In Tutorial

By: RickyRombo

Things you need

- A host or a server that can:
 - Send emails
 - Host mySQL databases
 - Function PHP
(check out Byethost.com – they offer all this for FREE and I use them for this tutorial)
- The Get object (which can be found on [Clickteam's Released Extension forum](#) or on Aquadasoft.com)
- Multimedia Fusion 2
- Notepad or equivalent (such as Notepad++)

NOTE: Even if you already have the Get object, redownload just in case you don't have the newest version!!!

Introduction

So you're making a game, and you decide it's going to be like an RPG or some sort of long term game. You make it an online arcade game or put a download online, and realize – wouldn't it be AWESOME if the user could load their data from ANY computer?? Or maybe you're making a Flash website with MMF2's Flash Exporter, and you realize you want people to be able to leave comments – wouldn't it be AWESOME if you could make them use accounts so you don't get spammed or false identities??

Whatever the reason, you clicked the link to this tutorial. Most people, when it comes to storing information online, immediately turn to the “writing an INI to a server” idea. BUZZ – this is wrong. This will slow down your server massively, not to mention leave a HUGE security issue with file transferring and such. The basic idea you need to get into your head is PHP + mySQL.

When you make a PHP file, the code is hidden from viewers. They cannot go to “view source”. This is one of my favorite features of PHP, although I'm a bit of a PHP newbie – as far as features. The script in this zip can connect to a mySQL database and read and write to it, without the information about the server and database being leaked out.

Setting up the Database

Enough about the coolness of PHP, lets set up our database. If you're using Byethost, these instructions will probably fit perfectly. Log in to the control panel, and scroll down to the “Database Management” section. Click on “MySQL Databases”. Create a new database (unless you want to use an existing one) and call it whatever you want.

atabases.

Create a NEW database::

<database-name>

Create Database

Now, on the left side it should show your database. Copy the name

Database
Name:  loginTutorial
[Backup] [Admin]

and place it in Notepad or something for later. Take the WHOLE name, not just the name you typed in. In Byethost, the first part is your username (I blotched mine out)

After you have that put somewhere, click where it says “Admin” under the name. It will take you to php-myadmin to edit the content of the database. It should show a dialog for creating a table. Type in a name for your table, and keep it with the database name for later. Where it says “Number of fields:” type 5 for now. Click “Go”.

Don't be overwhelmed. This is just the setup for the five fields. The first field will be called “name”. Give it a type of “VARCHAR”, which means it is a combination of characters with a variable length, and give it a length of 20, which limits usernames to 20 characters. If you think you want more than that, go on and make the number higher. Also, set the value in the “Default” column to “As defined:”.


The next field will be called “password”. Give it a type of “VARCHAR” again, but limit this one to 40 in the length. This will have to be 40, or the php won't work. I know that users don't usually have a 40 character password, but their password isn't what goes here – atleast, not exactly. We'll talk more on that later.

The field after that will be the user's email, so just call it “email”. Again give it the type of “VARCHAR” and limit it however you feel. I would NOT suggest 20 but that's what I did... some users probably have longer emails. DO NOT SET TO 20! SET TO 64 (AT LEAST)

The last two fields are the only ones that won't be “VARCHAR” The first one we'll call “validated”. This will test to see if the user verified that their email works. Give it a type of “TINYINT”. It's length will be 1. All we need is a few numbers – 0 means it hasn't been validated, 1 means it has, and -1 means that someone received the email by mistake and has deactivated the account.

The very last field will be called “creationdate”. Set the type to “TIMESTAMP” and don't give it a length. Instead, go to the “Default” column and give it the setting “CURRENT_TIMESTAMP”. This will get the date that the user registered. This way, you could make a php script to check if the user hasn't been validated for two days and delete it. We won't do that in this tutorial, but it's here in case you want to set that up. You could also manually check the database for accounts that haven't been validated for a certain amount of time and delete them.

When you're done, it should look something like this:

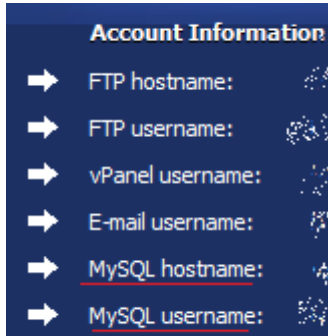
Field	Type 	Length/Values ¹	Default ²
name	VARCHAR	20	As defined:
password	VARCHAR	40	None
email	VARCHAR	20	None
validated	TINYINT	1	None
creationdate	TIMESTAMP		CURRENT_TIMESTAMP

If it does, press “Save”. It should give you a notice that says “No index defined!”. To fix this, hover over the icons under “Actions:” and click on the one that says “Index” for the name field.

That's it! Now let's move on to the PHP!

Setting up the PHP

Open up the PHP in this zip in Notepad or whatever you're using. All you need to fill in is the top – under where it says //CHANGE THE INFORMATION BELOW



Pretty self explanatory, but yeah. To find out your server information, go to the control panel again. If you're in Byethost, it will be in the bottom left. You want the MySQL Hostname and the MySQL Username. The password should be the same as your VistaPanel password.

The database you want to connect to is the one we created before. Same with the name of the table. The \$code and the \$hash are both used to add security to the PHP. Make them unique and very weird so they're harder to crack.

The rest is pretty much stuff for the emails. \$yourEmail is the email that shows up in the “From:” part of the message. \$gameName is the name that will show up in the subject, as in “Validation code for \$gameName” or “Account information for \$gameName”. The name to display will be at the end of the message. It doesn't have to have the game name in there, it's just an easy way of making a name. The \$pageLocation is the url of the php. Wherever you upload the php, that's what you want to put there. So if you named the php “myLogIn.php” and uploaded it to “<http://yoursite.com>” in a folder called “/coolstuff/”, you'd put “<http://yoursite.com/coolstuff/myLogIn.php>” for the pageLocation. That's it for the PHP!

Linking to the application

Now all you need to do is make your application do all the tasks! Open up the template.mfa in this zip. Each frame of the Application (except for “Game”) is named after a task the PHP can carry out. All you need to do is set the “task” to one of those five things, and add the required information (shown in the comments/code) to do that task. Note: Resend Information actually resets the password because the original password is not stored, but instead it stores the hash. Another thing: Limit the textbox sizes to the respective lengths. It'd be a shame for a user to sign up successfully and not be able to log in because the whole password doesn't match!

The template is pretty much self explanatory. Give it a good look over, and PM me, RickyRombo, on the forums If you need some help!