Pits Of Doom Lesson 10: Member Interaction

In the last lesson we added the ability to fight monsters on our map and added a graphic library to display our maps and updated the map editor so we can easily customize our maps and how they look from here on out. In this lesson we'll discuss member interaction, how to build a chat room, and how to display other member's characters on the map when they're logged into the game.

Lesson Concepts

No matter what you may think there is no such thing as "real time" as a result of how the Internet works. Let's take visiting your favorite website for example.

On your computer you open a browser, type in the domain name and hit enter or press the go button. When this happens your computer sends a packet of information out to your Internet connection (or wireless router) and then waits for a response. Your request is passed from place to place until it finds a server that's waiting and listening for your requests. When the server gets your request it processes it and sends back a result to your computer. So when you see the google homepage you've actually sent a request, the google server has heard your request, generated a response, and then your computer hears the response and displays the information from the google server on your browser screen. This is also why people who have dial up experience slow Internet.

So the question becomes, how do you hide or overcome delays that result from Internet communications? Some features of our Pits of Doom game are inherently perfect for covering up this problem. Take a chat room for instance. When you use a chat room you expect there will be a delay before you get your response. In this case the delay is to some degree expected by the user.

Now take our map. If we see another member on the same map level we don't expect it to jump from place to place as it moves around. Instead it should seamlessly navigate around the map even if we're not moving ourselves.

Creating The Chat Room

The concept behind a chat room is simple. Every time someone posts a message we store it to the database. In the chat screen we always select the last few messages inserted into the database. Finally we update the chat display every couple of seconds so it appears as if people are talking simultaneously.

Let's start by making a table for the chat messages. We'll need a way to identify which message belongs to which member, a place to store their text, and then the date/time they posted the message to make searching and sorting easier if we ever want to reference a particular day/time in the chat log.

```
mysql > CREATE  TABLE  chatroom (
   id INT NOT  NULL  AUTO_INCREMENT,
```

```
mid INT NOT  NULL,
message MEDIUMTEXT NOT  NULL,
date DATETIME NOT  NULL,
PRIMARY  KEY (id),
UNIQUE (id);
```

There are two fields in this table we haven't talked about before. The first one is mediumtext. This is one of three text options available in MySQL. There's tinytext, mediumtext, text, and longtext. These store varying amounts of data respectively. For everything we're doing with this game we'll only need a mediumtext or a text field type as those are the most common for the amount of data we're going to store. So the question is, why not always use the text or longtext options? MySQL has to allocate space for the information that would fit in the different sized text fields. By using the smallest one we need we'll increase our database speeds and cut down on any potentially wasted space.

Implementing The Chat Room

This is a fairly simple process. We have to tie the message box to the chat message submit button. When someone presses the submit button and they have a message we're going to add that message to the chat table.

How we display the chat messages is a bit more complicated. We have two options:

1. Use PHP to automatically refresh the screen periodically.
2. Use Ajax to re-generate the screen without refreshing the whole page (best solution).

Since this is a PHP/MySQL game we're going to go with the first PHP option so you don't have to know any Ajax. If you want the Ajax version you can contact me and purchase it as an addon for $25. Our script to display the most recent chat room messages looks like this:

```
<?php
/*****************
* File: chattext.php
* Date: 6.17.2011
* Author: design1online.com, LLC
* Purpose: display chat text messages
******************/
require_once('../oop/mysqlobj.php');
require_once('functions.php');
require_once('dbconnect.php');
$MAX_MESSAGES = 25;
$REFRESH_SECONDS = 5;

echo " <html> <head> <meta http-equiv=\"Content-Type\" content=\"text/html\" /> <link rel=\"stylesheet\"
type=\"text/css\" href=\"../css/default.css\" /> <meta http-equiv=\"refresh\"
content=\"$REFRESH_SECONDS\"> </head> <body class=\"chat\">";
```

```
$loop = mysql_query("
    SELECT
        C.mid,
        M.username,
        C.message,
        C.date
    FROM
        chatroom C
    INNER JOIN members M ON M.id = C.mid
    ORDER BY
        C.id DESC
    LIMIT $MAX_MESSAGES")
    or die ('Cannot load chat room messages ' . mysql_error());

 while ($row = mysql_fetch_assoc($loop))
    echo "<span class=\"datetime\">" . date('g:ia', strtotime($row['date'])) . "</span>" .
htmlentities($row['username']) . ": " . htmlentities($row['message']) . "<br/>";

 echo "</body>
    </html>";
```

Displaying Other Members

Not only do we want to talk to other members we want to see them on the map. In order to do this we'll need to check to see if the online flag is set to true. Whenever we login using our login form we automatically set this value to true. The only problem we have now is how to determine if they're still playing or not.

Therefore we're going to create a new script called online.php and call it every time someone moves their character around the board. We'll use this script to automatically logout anyone who hasn't been active recently and restore the online status to anyone whose currently still playing. So in a nutshell online.php has to do two things:

1. Remove the online status from any player whose last login is more than 20 minutes ago
2. Update the member's last login date to the current date and time any time they interact with the game

```
/*******
* Check to see if there is another character in this spot
*******/
 function hasMember($x, $y, $z, $mapid) {
     $result = $_SESSION['database']->query("SELECT M.id FROM characters C
          INNER JOIN members M ON M.id = C.memberid
          WHERE C.active=1 AND M.online=1 AND C.x='$x' AND C.y='$y' and C.z='$z'
          AND C.mapid = '$mapid' LIMIT 1");
```

```
    $row = mysql_fetch_assoc($result);
    return $row['id'];
}
```

Once we've established a way to determine which members are still online and which members are currently offline we can modify our displayMap function to show other members currently on the map if there is no monster present. In order to do this we first have to find all active characters (remember they can have more than one character eventually) then check to make sure the owners of those characters are online. Once we know they're online we can check the character's coordinates to figure out whether or not they are on a specific map location.

Game Files

This lesson we implemented a chat room, created a script to update whether or not a member is currently logged in and wrapped it up by displaying those member's characters on our map.

Play The Working Version!

Sign into Pits of Doom using the login below or create an account today! To really see this lesson in action it's best if you open two different browsers and login to each of them (using the same browser with two windows won't work!) to see other character's that are current logged into the game.

Account 1	Account 2
Username: test Password: test	Username: test2 Password: testing

Download The Source Code

Use the view source utility or download a .zip archive

Lesson Concepts

Member interaction is a huge part of an online game with any kind of social or interactive quality about it. In pits of doom we'd like to be able to trade/buy weapons and fight other members so it's important that they be able to communicate and see other characters on the game world. In the next lesson we'll look at updating our chat room so you can click on another member's name to view more about their characters and send them private messages.

Find more at http://design1online.com!