

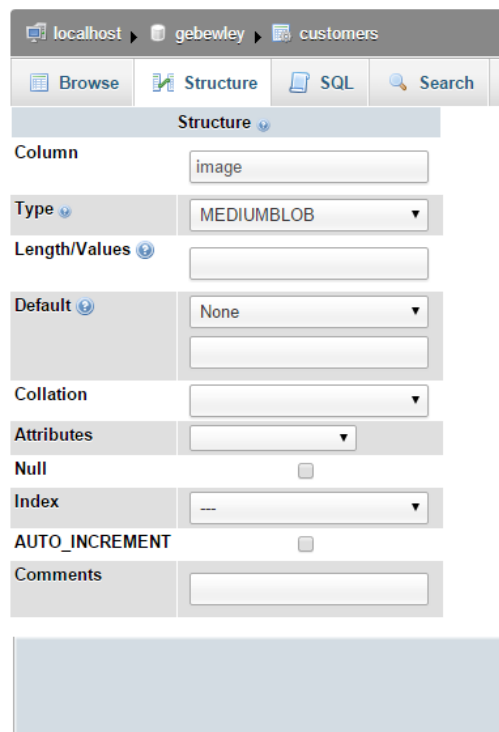
Adding File Upload to a CRUD Application

By Graham Bewley

This tutorial assumes that you already have a basic working CRUD Application. If you do not, you can follow the tutorial here: <http://www.startutorial.com/articles/view/php-crud-tutorial-part-1> (make sure to complete parts 1-3 of the tutorial).

This will teach you to upload files to a database. There are different types of file that you can upload to a database, but we will be using images in this demonstration.

The first step is to access your database and add a column to the table that you would like to work with. For demonstration purposes, we will be adding a column to the “customer” table. Click on the “Structure” tab of the customer table and click the button to add a column at the end of the table. We will call this column “image” and its type will be “MEDIUMBLOB” (BLOB stands for Binary Large Object). The rest of the fields can be left unchanged.



The screenshot shows a database management interface with the following fields and values:

Field	Value
Column	image
Type	MEDIUMBLOB
Length/Values	
Default	None
Collation	
Attributes	
Null	<input type="checkbox"/>
Index	---
AUTO_INCREMENT	<input type="checkbox"/>
Comments	

Next, we will be adding the image upload to the Create page of our CRUD application. Since we now have an extra column in our table, we must update the PHP portion of our Create page to reflect that change. In the section of code that triggers upon valid input (`if $valid{ . . . }`), replace what is there with the following code:

```

//If there is an image in the image space
if($_FILES['userfile']['size']>0) {
    ini_set('file-uploads',true);

    $fileName = $_FILES['userfile']['name'];
    $tmpName = $_FILES['userfile']['tmp_name'];
    $fileSize = $_FILES['userfile']['size'];
    $fileType = $_FILES['userfile']['type'];
    $fileType = (get_magic_quotes_gpc()==0
? mysql_real_escape_string($_FILES['userfile']['type'])
: mysql_real_escape_string(stripslashes ($_FILES['userfile'])));

    $fp = fopen($tmpName, 'r');
    $content = fread($fp, filesize($tmpName));
    $content = addslashes($content);
    fclose($fp);
    if (! get_magic_quotes_gpc() )
    {
        $fileName = addslashes($fileName);
    }

    $pdo = Database::connect();
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $sql = "INSERT INTO customers (name,email,mobile,image) VALUES ('$name',
'$email', '$mobile', '$content')";

    $q = $pdo->prepare($sql);
    $q->execute(array($name,$email,$mobile,$content));

    Database::disconnect();

    header("Location: people.php");
}

//Otherwise, there is no image in the image space
else {
    $pdo = Database::connect();
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $sql = "INSERT INTO customers (name,email,mobile) values(?, ?, ?)";

```

```

        $q = $pdo->prepare($sql);

        $q->execute(array($name,$email,$mobile));

        Database::disconnect();

        header("Location: people.php");

    }

```

This code will upload an image to the table if there is an image selected by the user. Otherwise, the customer will be added without an image.

Notice near the top of the code block, where `$content` is instantiated. This is the actual image file that we are uploading. The remaining variables (`$fileName`, `$fileSize`, etc.) are not necessary for the lightweight implantation of file upload that we are working on, but they will be left in. These could be used in other ways, for instance if we wanted to store names for our images, or sort images by file size on a page.

Note: you may need to change the code to reflect you're the index page for your CRUD application. The index page in this code is "people.php". You may also need to change the table name in the SQL statements if your table is called anything other than "customers".

Next up, we will be adding the file browser and button to our HTML. Underneath the input boxes for Name, Email, and Phone (but before the Create button), add the following code:

```

<div class="control-group" <?php echo !empty($pictureError)?'error':'';?>">

    <label class="control-label">Select An Image</label>

    <div class="controls">

        <input type="hidden" name="MAX_FILE_SIZE" value="16000000">

        <input name="userfile" type="file" id="userfile">

    </div>

</div>

```

This code adds the file browser to our application, storing the selected file as "userfile". If you navigate to the earlier block of code provided, you'll notice that this "userfile" is what's used to separate the file into attributes like `fileName`, `fileSize`, etc.

We now have an input box for selecting a file, and the PHP code for storing that file in our table. To test the file upload, you can refresh the Create page of your CRUD application and use the file upload button to add an image to a new customer. After that, there are a number of ways to check that our file upload was successful. The first, and easiest of which is to refresh your database and ensure that your new image (MEDIUMBLOB) column now includes a file with some size greater than 0. Notice below that each of the entries in the customers database has something stored in their image attribute.

<div>←T→</div>									id	name	email	mobile	image
<input type="checkbox"/>		Edit		Inline Edit		Copy		Delete	18	Guy name	guy@email.com	12345	[BLOB - 548.1KiB]
<input type="checkbox"/>		Edit		Inline Edit		Copy		Delete	19	George Corser	george@corser.com	12345	[BLOB - 762.5KiB]
<input type="checkbox"/>		Edit		Inline Edit		Copy		Delete	20	Graham	graham@bewley.com	9898989	[BLOB - 858.8KiB]

Another way to check for an uploaded file is to view it in our application. This is the more practical approach, since the point of uploading an image is so that we can use it to distinguish a customer. We will view our new customer image in the Read page of our application.

The SQL statement used in the Read page should be fine as it is, since the statement selects everything from the customers database (`SELECT * FROM customers...`). The only thing we need to add to our page is some HTML code that displays the image. Underneath the other lines for displaying data in the HTML, add a div that shows our image:

```
<div class='control-group col-md-6'>
    <div class="controls ">
        <?php echo '';
        ?>
    </div>
</div>
```

This div will display our image, as long as the database column for customers which holds the image is called “image”. The less obvious looking code is converting the file into base 64. We have to do this in order to read the binary object’s code to display the image.

Now your refreshed Read page should be showing the image for the selected customer.