**Converting and Storing Images**

When the server receives the uploaded image file from the HTML form, it places it in a temporary location on the server. Your PHP code needs to retrieve the temporary file and push it into the products table BLOB data field. This requires a few different PHP features that you need to learn.

PHP provides the $\_FILES[] array variable, which contains information about any files uploaded using the file input type. The $\_FILES[] array is a multidimensional associative array that uses the format:

$\_FILES[*name*][*element*]

The *name* index matches the name value used in the file input type. This is called *picture* in the newproduct.inc.php code. So that's the value you need to use to extract the image file from the server.

Each uploaded file has several *element* indexes that define specific features of the uploaded file.

|  |  |
| --- | --- |
| **The $\_FILES[] array elements** | |
| **Element** | **Description** |
| name | The original name of the file on the visitor's PC. |
| type | The MIME type of the file. |
| size | The size (in bytes) of the file. |
| tmp\_name | The name of the temporary file created on the Web server. |
| error | The error code associated with the file upload. |

To retrieve the temporary filename for the image uploaded from the file input type named *picture*, you would access the $\_FILES['picture']['tmp\_name'] array element.

The addproducts.inc.php code uses the following code to retrieve the image file:

$thumbnail = getThumb($\_FILES['picture']);

This code passes the entire $\_FILES['picture'] array to a PHP function called getThumb() for processing. You need to create this file next. Just follow these steps:

1. Create a file called getThumb.php in the *mylibrary* folder you created under your store folder.
2. Enter the following code into the file:

<?php

function getThumb($Original)

{

if (!$Original['name'])

{

//no image supplied, use default

$TempName = "images/noimage.jpg";

$TempFile = fopen($TempName, "r");

$thumbnail = fread($TempFile, fileSize($TempName));

} else

{

//get image

$Picture = file\_get\_contents($Original['tmp\_name']);

//create image

$SourceImage = imagecreatefromstring($Picture);

if (!$SourceImage)

{

//not a valid image

echo "Not a valid image\n";

$TempName = "images/noimage.jpg";

$TempFile = fopen($TempName, "r");

$thumbnail = fread($TempFile, fileSize($TempName));

} else

{

//create thumbnail

$width = imageSX($SourceImage);

$height = imageSY($SourceImage);

$newThumb = imagecreatetruecolor(80, 60);

//resize image to 80 x 60

$result = imagecopyresampled($newThumb, $SourceImage,

0, 0, 0, 0,

80, 60, $width, $height);

//move image to variable

ob\_start();

imageJPEG($newThumb);

$thumbnail = ob\_get\_contents();

ob\_end\_clean();

}

}

return $thumbnail;

}?>

1. Save the file and exit the editor.
2. Edit the original admin.php file in the store folder.
3. After the include() statement for the login.php file, add a new include() function to include the getThumb.php function code file. It should now look like this:

include("/mylibrary/login.php");

include("/mylibrary/getThumb.php");

1. Save the file and exit the editor.

The job of the getThumb() function is to provide a small thumbnail version of the image the manager uploads. The getThumb() function first checks the *name* element of the uploaded file to see if the manager supplied a file for the image file.

If the manager didn't specify a file, you'll need to supply a generic image that you can use in the Web page:

1. Create a folder called *images* under the admin folder in your store application folder.
2. Take the *noimage.jpg* file you created back in Lesson 2 and copy it to the images folder.

The getThumbs() function reads this image using the fread() PHP function and will utilize it if the manager doesn't specify an image or if it can't convert the image the manager does specify.

If the manager specified a file image, the getThumbs() function uses the *file\_get\_contents()* function to read the temporary file on the server (specified using the *tmp\_name* element) into a PHP string variable, then attempts to create an image from the image string. If that's successful, it uses the GD2 library functions to resample the image to an 80 x 60-pixel thumbnail. This ensures that all of our product images are the same size, making our catalog layout better.

After creating the new thumbnail, the getThumbs() function code needs to place it back into a PHP variable to pass to the calling program. This gets a little tricky. You'll notice the small section of code that performs this function:

ob\_start();

imageJPEG($newThumb);

$thumbnail = ob\_get\_contents();

ob\_end\_clean();

Unfortunately, there's no function that converts an image back into a string value that you can pass to the calling program. You have to use some PHP tricks to do that.

The *ob\_start()* function creates an output buffer that stores anything that would've normally been output from the PHP code. Once the PHP sees the ob\_start() function, it creates a temporary buffer area in memory and redirects all output to that buffer instead of sending the output directly to the client browser. The imageJPEG() function normally outputs the image to the browser, but now, PHP redirects that output to the buffer area.

The *ob\_get\_contents()* function retrieves the contents of the buffer area. You use this to store the image value into a string PHP variable. To stop the buffering, you must use the *ob\_end\_clean()* function. After you close the output buffer, the output from the PHP code returns to normal and goes to the client's browser.

Now, the newly formed image (or our *no image available* image) is in the $thumbnail variable. The function then returns it back to the calling program.

Now back in the addproduct.inc.php file, you need to use the mysql\_real\_escape\_string() function on the thumbnail image in case there are any stray quotes or backslashes in the binary data. Then you're ready to use the INSERT statement to push in your new product.

Wow, that was a lot of work to do in one day! Let's move on to the Summary and wrap things up.