**Exercise 02\_05\_01 – Step 1**

In this Exercise, we will learn how to handle some of the tasks that PHP was particularly designed to do, work with data sources.



1. Create a folder named Exercise 02\_05\_01 and open it with your IDE. Create a new file called ***ViewFiles.php***. Scaffold a basic HTML code layout into it. Complete our standard opening documentation in the ***<head>*** element. Make sure to have the ***modernizr*** <script> linked in. Set the <title> content to ***View Files***:  
   ***<!DOCTYPE html>  
   <html lang="en">  
   <head>  
    <meta charset="utf-8">  
    <meta name="viewport" content="width=device-width">  
    <title>View Files</title>  
    <script src="modernizr.custom.65897.js"></script>  
   </head>  
   <body>  
   </body>  
   </html>***
2. In the <body>, create an ***<h2>*** element with content ***View Files***. Create a set of PHP standard script delimiters.  
    ***<h2>View Files</h2>  
    <?php  
     
    ?>***Copy the project folder into the appropriate spot on your Web Server and test it.
3. Insert the following code into the script section to read the files in a directory. Change the ***$dir*** value to be valid for your file system:  
    <?php  
    ***$dir = "../Exercise 02\_01\_01";  
    $openDir = opendir($dir);  
    while ($curFile = readdir($openDir)) {  
    echo "$curFile<br>";  
    }  
    closedir($openDir);*** ?>  
   Give this a browser/server test.
4. Change the code to ignore the ***current*** directory and the ***parent*** directory:  
    while ($curFile = readdir($openDir)) {  
    ***if (strcmp($curFile, '.') !== 0 && strcmp($curFile, '..') !==   
    0) {*** echo "$curFile<br>";  
    ***}*** }  
   Give this a browser/server test.
5. Change the code to make each of the output files a ***hyperlink***:  
    if (strcmp($curFile, '.') !== 0 && strcmp($curFile, '..') !== 0) {  
    ***echo "<a href=\"$dir/$curFile\">  
    $curFile</a><br>\n";*** }  
   Give this a browser/server test.

**Exercise 02\_05\_01 – Step 2**



1. Copy the file ***ViewFiles.php*** to a file named ***ViewFiles2.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***View Files 2.*** In the <body>, create an ***<h2>*** element with content ***View Files 2***. Create a set of PHP standard script delimiters:  
    ***<h2>View Files 2</h2>  
    <?php  
     
    ?>***Give this a server/browser test.
2. Change the ***opendir()*** call to a ***scandir()*** call:  
    ***$dirEntries = scandir($dir);***
3. Modify the ***while*** loop to a ***foreach*** loop to read the array gotten from the ***scandir()*** call, and do the output. Make sure to remove the ***closedir()*** call:  
    ***foreach ($dirEntries as $entry) {  
    if (strcmp($entry, '.') !== 0 && strcmp($entry, '..') !== 0)   
    {  
    echo "<a href=\"$dir/$entry\">$entry</a><br>\n";  
    }  
    }***  
   Give this a server/browser test.

**Exercise 02\_05\_01 – Step 3**



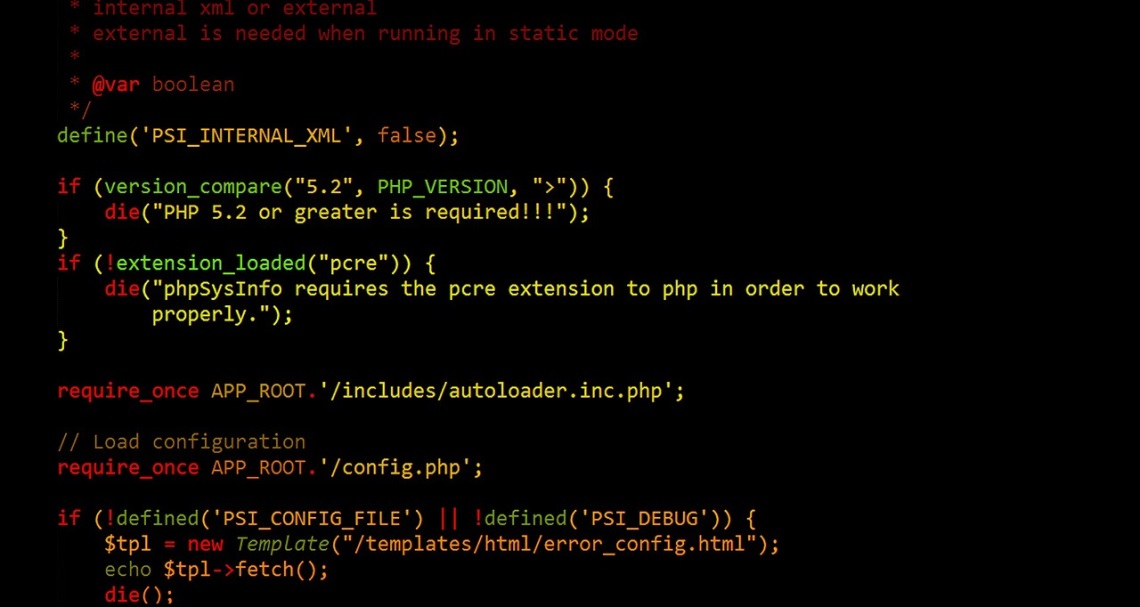
1. Copy the file ***ViewFiles2.php*** to a file named ***ViewFiles3.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***View Files 3.*** In the <body>, create an ***<h2>*** element with content ***View Files 3***. Create a set of PHP standard script delimiters:  
    ***<h2>View Files 3</h2>  
    <?php  
     
    ?>***Give this a server/browser test.
2. Under the ***scandir()*** call, let’s set up an HTML ***<table>*** to hold the directory information:  
    ***echo "<table border='1' width='100%'>\n";*** foreach ($dirEntries as $entry) {  
    if (strcmp($entry, '.') !== 0 && strcmp($entry, '..') !== 0) {  
    echo "<a href=\"$dir/$entry\">$entry</a><br>\n";  
    }  
    }  
    ***echo "</table>\n";***Give this a server/browser test.
3. Now let’s set up the table main heading and column headings. Notice of the ***htmlentities()*** function, used to convert any characters in a string that are applicable to the proper HTML entity representation:  
    echo "<table border='1' width='100%'>\n";  
    ***echo "<tr><th colspan='4'>Directory Listing for <strong>" .   
    htmlentities($dir) . "</strong></th></tr>\n";  
    echo"<tr>";  
    echo "<th><strong><em>Name</em></strong></th>";  
    echo "<th><strong><em>Owner</em></strong></th>";  
    echo "<th><strong><em>Permissions</em></strong></th>";  
    echo "<th><strong><em>Size</em></strong></th>\n";  
    echo"</tr>\n";***Give this a server/browser test.
4. We can now modify our foreach loop to place the information into the table by replacing the ***echo*** statement with the following code:  
    if (strcmp($entry, '.') !== 0 && strcmp($entry, '..') !== 0) {  
    ***$fullEntryName = $dir . "/" . $entry;  
    echo "<tr><td>";  
    if (is\_file($fullEntryName)) {  
    echo "<a href=\"$fullEntryName\">" .   
    htmlentities($entry)   
    . "</a>";  
    }  
    else {  
    echo htmlentities($entry);  
    }  
    echo "</td><td align='center'>" .   
    fileowner($fullEntryName);  
    if (is\_file($fullEntryName)) {  
    $perms = fileperms($fullEntryName);  
    $perms = decoct($perms % 01000);  
    echo "</td><td align='center'>0$perms";  
    echo "</td><td align='right'>" .   
    number\_format(filesize($fullEntryName), 0) . "   
    bytes";  
    }  
    else {  
    echo "</td><td colspan='2'   
    align='center'>&lt;DIR&gt;";  
    }  
    echo "</td><tr>\n";*** }  
   Give this a server/browser test.

**Exercise 02\_05\_01 – Step 4**



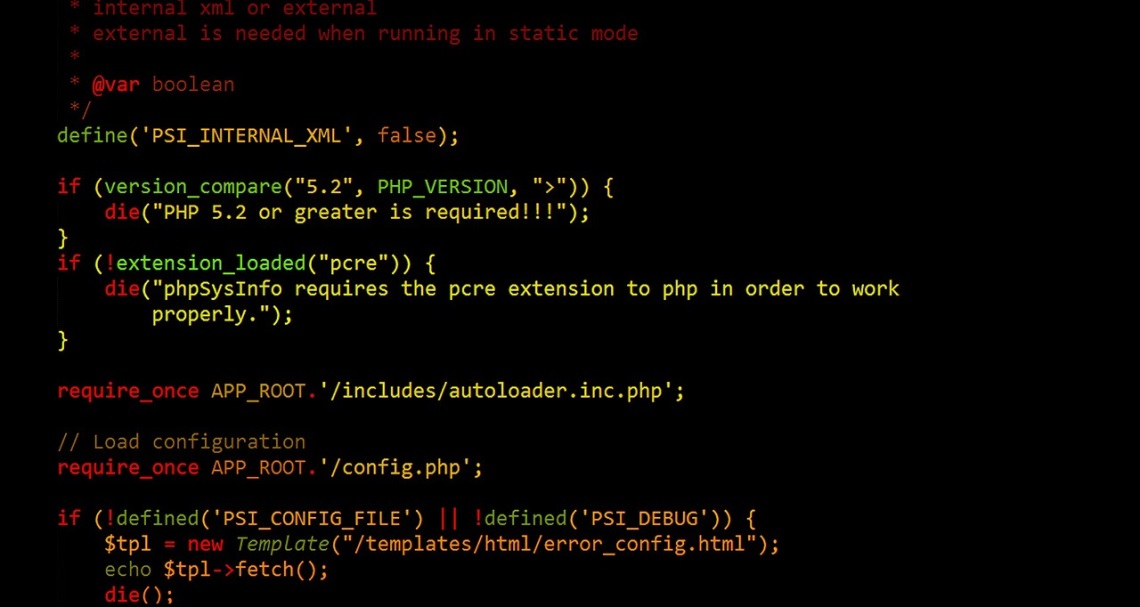
1. Copy the file ***ViewFiles.php*** to a file named ***FileUploader.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***File Uploader.*** In the <body>, create an ***<h2>*** element with content ***File Uploader***. Create a set of PHP standard script delimiters:  
    ***<h2> File Uploader </h2>  
    <?php  
     
    ?>***Give this a server/browser test.
2. First, let’s create our form to select a file for upload. Directly below the empty script delimiters, enter the following ***HTML*** code:  
    *<?php  
      
    ?>* ***<form action="FileUploader.php" method="post"   
    enctype="multipart/form-data">  
    <input type="hidden" name="MAX\_FILE\_SIZE"   
    value="25000">  
    File to upload:<br>  
    <input type="file" name="newFile"><br>  
    (25,000 byte limit)<br>  
    <input type="submit" name="upload" value="Upload the   
    File"><br>  
    </form>***Give this a server/browser test.

**Exercise 02\_05\_01 – Step 5**



1. Return to the ***FileUploader.php*** file. Let’s start to build our upload code. We will handle an error condition first, which will allow us to use what we have learned about retrieving file information. Add the following code to the script section to start handling the file ***upload***. We will test the error condition with a failing ***if*** statement:  
    *<?php* ***$dir = ".";  
    if (isset($\_POST['upload'])) {  
    if (isset($\_FILES['newFile'])) {  
    if (isset($\_FILES['oldFile'])) { // debug  
      
    }  
    else {  
    echo "There was an error uploading \"" .   
    htmlentities($\_FILES['newFile']['name']) .   
    "\".<br>\n";  
    }  
    }  
    }*** *?>*Give this a server/browser test to make sure the upload error is generated.

**Exercise 02\_05\_01 – Step 6**



1. Return to the ***FileUploader.php*** file and we can build out our upload code. Replace the code in the debug ***if*** statement with the following:  
    *if (isset($\_FILES['newFile'])) {* ***if (move\_uploaded\_file(  
    $\_FILES['newFile']['tmp\_name'], $dir . "/" .   
    $\_FILES['newFile']['name']) === true) {  
    chmod($dir . "/" . $\_FILES['newFile']['name'],   
    0644);  
    echo "File \"" . htmlentities(  
    $\_FILES['newFile']['name']) . "\"successfully   
    uploaded.<br>\n";***Give this a server/browser test to make sure the upload success message appears. Go back and run ***ViewFiles.php*** to see if it is there. Now purposely try to upload a file that is too big

**Exercise 02\_05\_01 – Step 7**



1. Copy the file ***ViewFiles.php*** to a file named ***FileDownloader.php***, and open it with your IDE. Remove all of the code from the <body> including the entire script area. Update the documentation and set the <title> content to ***File Download Error.***
2. At the very top of the file, place an empty set of PHP script delimiters. This will allow us to provide the appropriate headers to the server before any possibility of sending any HTML content  
    ***<?php  
     
    ?>***Give this a server/browser test.
3. Add the following HTML code to the <body> element to provide us with an error page. Make sure to include the very bottom PHP script:  
   <body>  
    ***<p>There was an error downloading "<?php echo   
    htmlentities($\_GET['filename']); ?>"</p>  
    <p><?php echo htmlentities($errorMsg); ?></p>***</body>  
   Give this a server/browser test. It should throw two errors because of the undefined PHP index and variable.
4. Enter the following code within the top script delimiters to ascertain if the requested file exists and is readable:  
   <?php  
   ***$dir = "../Exercise 02\_01\_01";  
   if (isset($\_GET['fileName'])) {  
    $fileToGet = $dir . "/" . stripslashes($\_GET['fileName']);  
    if (is\_readable($fileToGet)) {  
    $errorMsg = "";  
    $showErrorPage = true;  
    }  
    else {  
    $errorMsg = "Cannot read \"$fileToGet\"";  
    $showErrorPage = true;  
    }  
   }  
   else {  
    $errorMsg = "No filename specified";  
    $showErrorPage = true;  
   }  
   if ($showErrorPage) {***?>
5. Go to the bottom of the document and enter the following to close the last ***if*** statement:  
   </html>  
   ***<?php  
   }  
   ?>***Give this a server/browser test. It should ***first*** throw one error in downloading, that we actually sent in the HTML. A ***second*** error should show an undefined index, because we have nothing in the ***$\_GET*** superglobal array as yet. A ***third*** error is also shown by us from the PHP because there is no filename to download.
6. Now let’s give our downloader a file to download by putting a token into a URL that will call FileDownloader.php. Copy the file ***ViewFiles3.php*** to a file named ***ViewFiles4.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***View Files 4.*** In the <body>, create an ***<h2>*** element with content ***View Files 4***. Change the Replace the line with the ***<a>*** element with the following ***hyperlink***:  
   ***echo "<a href=\"FileDownloader.php?fileName=$entry\">" .   
    htmlentities($entry) . "</a>";***Run ViewFiles4.php and click one of the links. Our undefined index error should be gone because we have a filename in our URL and therefore the index is found in the $\_GET array.
7. Now let’s send our appropriate headers to the server to enable our download. We do this inside the **if (isreadable()** statement, making sure to set the ***$showErrorPage*** variable to ***false***:  
    if (is\_readable($fileToGet)) {  
    ***header("Content-Description: File Transfer");  
    header("Content-Type: application/force-download");  
    header("Content-Disposition: attachment; filename=\"" .   
    $\_GET['fileName'] . "\"");  
    header("Content-Transfer-Encoding: base64");  
    header("Content-Length: " . filesize($fileToGet));  
    readfile($fileToGet);  
    $showErrorPage = false;*** }  
   Give this a server/browser test.

**Exercise 02\_05\_01 – Step 8**



1. Make a folder below the current project folder named ***comments***. Copy the file ***ViewFiles.php*** to a file named ***VisitorComments.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Visitor Comments***. Create a set of PHP standard script delimiters:  
    ***<?php  
     
    ?>***Give this a server/browser test.
2. Add the following HTML code directly below the closing PHP script delimiter:  
    *?>* ***<h2>Visitor Comments</h2>  
    <form action="VisitorComments.php" method="post">  
    Your name: <input type="text" name="name"><br>  
    Your email: <input type="email" name="email"><br>  
    <textarea name="comment" rows="6"   
    cols="100"></textarea><br>  
    <input type="submit" name="save" value="Submit your   
    comment"><br>  
    </form>***Give this a server/browser test.
3. Begin to build the code to store the form data entered. The data will go into a string variable named ***$saveString***. Add the following to the PHP script, with a line of debug code:  
    *<?php* ***$dir = "./comments";  
    if (is\_dir($dir)) {  
    if (isset($\_POST['save'])) {  
    if (empty($\_POST['name'])) {|  
    echo "Unknown Visitor\n";  
    }  
    else {  
    $saveString = stripslashes($\_POST['name']) . "\n";  
    echo "\$saveString: $saveString<br>";  
    }  
    }  
    }*** *?>*  
   Give this a server/browser test. Make sure to submit an empty form, then with a name filled in.
4. Continue to build up the $***saveString*** variable by placing the following code above the debug ***echo*** statement. Note the use of the PHP date(‘r’) function, in which the parameter provides a fully formatted time stamp:  
    ***$saveString .= stripslashes($\_POST['email']) . "\n";  
    $saveString .= date('r') . "\n";  
    $saveString .= stripslashes($\_POST['comment']) .   
    "\n";*** echo "\$saveString: $saveString";  
   Give this a server/browser test. Make sure to submit an empty form, then with a name, email, and a comment filled in.
5. Now let’s build a filename to store it in and place it in a ***$saveFileName*** variable. We will use group of PHP timestamp functions to do this. The ***microtime()*** function will return a UNIX timestamp with microseconds. The PHP ***explode()*** function will break the string into an array. We then ***cast*** each of the array elements into floats, add them in reverse order, and store them in a variable ***$timeStamp***. Lastly, we will create the ***$saveFileName*** by putting all these pieces together with the word ***Comment*** and a ***.txt*** extension. We will place debug statements in all of this so that we can watch the transformations:  
    *echo "\$saveString: $saveString<br>";* ***$currentTime = microtime();  
    echo "\$currentTime: $currentTime <br>";  
    $timeArray = explode(" ", $currentTime);  
    echo var\_dump($timeArray) . "<br>";  
    $timeStamp = (float)$timeArray[1] +   
    (float)$timeArray[0];  
    echo "\$timeStamp: $timeStamp<br>";  
    $saveFileName = "$dir/Comment.$timeStamp.txt";  
    echo "\$saveFileName: $saveFileName<br>";***Give this a server/browser test.
6. We can now build the code that writes our ***$saveString*** to our ***$saveFileName*** and stores it on the server. But first, let’s make sure we have an error trap if we are not successful:  
    *echo "\$saveFileName: $saveFileName<br>";* ***if (false) {  
      
    }  
    else {  
    echo "There was an error writing\"" .   
    htmlentities($saveFileName) . "\".<br>\n";  
    }***  
   Give this a server/browser test.
7. Now let’s fill in the code in the ***if*** statement to do the save::  
    *echo "\$saveFileName: $saveFileName<br>";* ***if (file\_put\_contents($saveFileName, $saveString) >   
    0)   
    {  
    echo "File \"" . htmlentities($saveFileName) . "\"   
    successfully saved.<br>\n";  
    }*** *else {  
    echo "There was an error writing \"" .   
    htmlentities($saveFileName) . "\".<br>\n";  
    }*Give this a server/browser test. Go back to the server index to check that any files you wrote are there. Click one and you should see the text that was placed in it.

**Exercise 02\_05\_01 – Step 9**



1. Copy the file ***ViewFiles.php*** to a file named ***VisitorFeedback.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Visitor Feedback***. Create a set of PHP standard script delimiters:   
    ***<?php  
     
    ?>***Give this a server/browser test.
2. Add the following HTML code directly below the opening ***<body>*** element:  
   *<body>* ***<h2>Visitor Feedback</h2>  
    <hr>***Give this a server/browser test.
3. Add the following to the PHP script, using ***scandir()*** to retrieve the comment files:  
    *<?php* ***$dir ="./comments";  
    if (is\_dir($dir)) {  
    $commentFiles = scandir($dir);  
    foreach ($commentFiles as $fileName) {  
    if ($fileName !== "." && $fileName !== "..") {  
    echo "From <strong>$fileName</strong><br>";  
    }  
    }  
    }*** *?>*Give this a server/browser test.
4. Let’s build out the ***foreach*** loop to provide some decent output:  
    *echo "From <strong>$fileName</strong><br>";* ***echo "<pre>\n";  
    $comment = file\_get\_contents($dir . "/" .   
    $fileName);  
    echo $comment;  
    echo "</pre>\n";  
    echo "<hr>\n";***Give this a server/browser test.

**Exercise 02\_05\_01 – Step 10**



1. Copy the file ***VisitorFeedback.php*** to a file named ***VisitorFeedback2.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Visitor Feedback 2***. Create a set of PHP standard script delimiters:   
    ***<?php  
     
    ?>***Give this a server/browser test.
2. Add the following HTML code directly below the opening ***<body>*** element:  
   *<body>* ***<h2>Visitor Feedback 2</h2>  
    <hr>***Give this a server/browser test.
3. Replace the ***file\_get\_contents()*** call in the PHP script with the ***readfile()*** call. Remove the ***echo*** call that displays ***$contents***. This is the preference when you do not need to save the file contents in a variable:  
    *echo "<pre>\n";* ***readfile($dir . "/" . $fileName);*** *echo "</pre>\n";  
    echo "<hr>\n";*Give this a server/browser test.

**Exercise 02\_05\_01 – Step 11**



1. Copy the file ***VisitorFeedback.php*** to a file named ***VisitorFeedback3.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Visitor Feedback 3***. Create a set of PHP standard script delimiters:   
    ***<?php  
     
    ?>***Give this a server/browser test.
2. Add the following HTML code directly below the opening ***<body>*** element:  
   *<body>* ***<h2>Visitor Feedback 3</h2>  
    <hr>***Give this a server/browser test.
3. Replace the code in the PHP script with the following, starting with the opening ***<pre>*** element and including the closing ***</pre>*** element:  
    *echo "From <strong>$fileName</strong><br>";* ***$comment = file($dir . "/" . $fileName);  
    echo "From: " . htmlentities($comment[0]) .   
    "<br>\n";  
    echo "Email Address: " . htmlentities($comment[1]) .   
    "<br>\n";  
    echo "Date: " . htmlentities($comment[2]) . “  
    "<br>\n";  
    $commentLines = count($comment);  
    for ($i = 3; $i < $commentLines; $i++) {  
    echo htmlentities($comment[$i]) . "<br>\n";  
    }*** *echo "<hr>\n";*Give this a server/browser test.

**Exercise 02\_05\_01 – Step 12**



1. Copy the file ***VisitorComments.php*** to a file named ***VisitorComments2.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Visitor Comments 2***. Create a set of PHP standard script delimiters:   
    ***<?php  
     
    ?>***Give this a server/browser test.
2. Add the following HTML code directly below the opening ***<body>*** element:  
   *<body>* ***<h2> Visitor Comments 2</h2>  
    <hr>***
3. Update the opening ***<form>*** element ***action*** attribute:  
   ***<form action="VisitorComments2.php" method="post">***Give this a server/browser test.
4. Replace the entire ***file\_put\_contents() if…else*** in the PHP script with the following incremental write code. Notice the ***fopen()*** argument ***wb***, which opens the file and overwrites it, or creates it if it does not exist. The b opens it in ***binary*** mode:  
    *echo "\$saveFileName: $saveFileName<br>";* ***$fileHandle = fopen($saveFileName, "wb");  
    if ($fileHandle === false) {  
    echo "There was an error creating \"" .   
    htmlentities($saveFileName) . "\".<br>\n";  
    }  
    else {  
    if (fwrite($fileHandle, $saveString) > 0) {  
    echo "Successfully wrote to file \"" .   
    htmlentities($saveFileName) . "\".<br>\n";  
    }  
    else {  
    echo "There was an error writing to \"" .   
    htmlentities($saveFileName) . "\".<br>\n";  
    }  
    fclose($fileHandle);  
    }***Give this a server/browser test. Check that the file is correct in the ***comments*** subdirectory.

**Exercise 02\_05\_01 – Step 13**



1. Copy the file ***VisitorComments2.php*** to a file named ***VisitorComments3.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Visitor Comments 3***. Create a set of PHP standard script delimiters:   
    ***<?php  
     
    ?>***Give this a server/browser test.
2. Add the following HTML code directly below the opening ***<body>*** element:  
   *<body>* ***<h2> Visitor Comments 3</h2>  
    <hr>***
3. Update the opening ***<form>*** element ***action*** attribute:  
   ***<form action="VisitorComments3.php" method="post">***Give this a server/browser test.
4. Place the ***if (fwrite()*** statement inside another ***if*** statement that ***locks*** the file, then ***unlocks*** it when the write operation is done:  
    ***if (flock($fileHandle, LOCK\_EX)) {*** *if (fwrite($fileHandle, $saveString) > 0) {  
    echo "Successfully wrote to file \"" .   
    htmlentities($saveFileName) . "\".<br>\n";  
    }  
    else {  
    echo "There was an error writing to \"" .   
    htmlentities($saveFileName) . "\".<br>\n";  
    }* ***flock($fileHandle, LOCK\_UN);*** *}  
    fclose($fileHandle);*
5. Add an ***else*** statement to the ***if (flock()*** statement to cover if the file lock can not be obtained: ***else {  
    echo "There was an error locking file \"" .   
    htmlentities($saveFileName) . "\" for   
    writing.<br>\n";  
    }***Give this a server/browser test. Check that the file is correct in the ***comments*** subdirectory.

**Exercise 02\_05\_01 – Step 14**



1. Copy the file ***VisitorFeedback3.php*** to a file named ***VisitorFeedback4.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Visitor Feedback 4***. Create a set of PHP standard script delimiters:   
    ***<?php  
     
    ?>***Give this a server/browser test.
2. Add the following HTML code directly below the opening ***<body>*** element:  
   *<body>* ***<h2>Visitor Feedback 4</h2>  
    <hr>***Give this a server/browser test.
3. Replace the ***file()*** statement with the following ***fopen()*** call and use an ***if*** statement as an error trap. Enclose all the rest of the statements with an ***else*** clause, putting an fclose() as the last statement of the ***else*** clause:  
    echo "From <strong>$fileName</strong><br>";  
    ***$fileHandle = fopen($dir . "/" . $fileName, "rb");  
    if ($fileHandle === false) {  
    echo "There was an error reading file   
    \"$fileName\".<br>\n";  
    }  
    else {*** echo "From: " . htmlentities($comment[0]) . "<br>\n";  
    echo "Email Address: " . htmlentities($comment[1]) .   
    "<br>\n";  
    echo "Date: " . htmlentities($comment[2]) . "<br>\n";  
    $commentLines = count($comment);  
    for ($i = 3; $i < $commentLines; $i++) {  
    echo htmlentities($comment[$i]) . "<br>\n";  
    }  
    echo "<hr>\n";  
    ***fclose($fileHandle);  
    }***
4. Modify the code in the PHP script that displays the file contents with ***fgets()*** statements as follows:   
    *else {* **$from = fgets($fileHandle);  
    echo "From: " . htmlentities($from) . "<br>\n";  
    $email = fgets($fileHandle);  
    echo "Email Address: " . htmlentities($email) .   
    "<br>\n";  
    $date = fgets($fileHandle);  
    echo "Date: " . htmlentities($date) . "<br>\n";  
    $comment = "";  
    while (!feof($fileHandle)) {  
    $comment .= fgets($fileHandle);  
    }  
    echo htmlentities($comment) . "<br>\n";** *echo "<hr>\n";  
    fclose($fileHandle);  
    }*Give this a server/browser test.

**Exercise 02\_05\_01 – Step 15**



1. Copy the file ***ViewFiles.php*** to a file named ***BackupComments.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Backup Comments***. Create a set of PHP standard script delimiters:   
    ***<?php  
     
    ?>***Give this a server/browser test.
2. Declare the following two global variables to set up the ***source*** and ***destination*** directories:  
    *<?php* ***$source = "./comments";  
    $destination = "./backups";*** *?>*
3. Set up an error trap to make sure the destination directory exists:  
    ***if (!is\_dir($destination)) {  
    echo "The backup directory \"$destination\" does not   
    exist.\n";  
    }  
    else {  
      
    }***Give this a server/browser test without creating the ***backups*** directory. Make a folder below the current project folder named ***backups*** and retest.
4. Now let’s start to build out our backup functionality in the ***else*** statement. Starting with an error trap to make sure the source directory exists, with some debug code:  
    *else {* ***if (is\_dir($source)) {  
    echo "The source directory \"$source\" does exist.\n";  
    }  
    else {  
    echo "The source directory \"$source\" does not   
    exist.\n";  
    }*** *}*Give this a server/browser test.
5. Now let’s finish building our backup functionality in the ***if*** statement if the source exists. Replace the ***debug echo*** statement with the following code:  
    *if (is\_dir($source)) {* ***$totalFiles = 0;  
    $filesMoved = 0;  
    $dirEntries = scandir($source);  
    foreach ($dirEntries as $entry) {  
    if ($entry !== "." && $entry !== "..") {  
    ++$totalFiles;  
    if (copy("$source/$entry",   
    "$destination/$entry")) {  
    ++$filesMoved;  
    }  
    else {  
    echo "Could not move file \"" .   
    htmlentities($entry) . "\".<br>\n";  
    }  
    }  
    }  
    echo "<p>$filesMoved of $totalFiles files successfully   
    backed up.</p>\n";*** *}*Give this a server/browser test. Check that the files are correct in the ***backups*** subdirectory.