|  |
| --- |
| Министерство образования Республики Беларусь  УО «Полоцкий государственный университет им. Ефросинии Полоцкой» |

|  |  |
| --- | --- |
|  | Факультет информационных технологий  Кафедра технологий программирования |

|  |
| --- |
| ОТЧЕТ |
| **к лабораторной работе №14(5)**  **Дисциплина**: «Программирование для Интернет» |

|  |  |
| --- | --- |
| Выполнил: | Студент группы 19-ИТ-2  Гилимович А.Н. |

|  |  |
| --- | --- |
| Проверила: | Преподователь  Виноградова А.Д. |

|  |
| --- |
| Полоцк, 2022 г. |

**Обработка исключений, работа с файловой системой при помощи языка**

**PHP.**

**Цель работы:** ознакомиться с конструкцией try catch finally, на практике понять принцип генерации исключений, разобраться в таких процессах как «чтение и запись файлов», «управление файлами и каталогами», «блокировка файлов, функция flock».

**Обработка исключений.**

**Задание 1.** Написать свой обработчик исключений, с выводом текста файла, где произошла ошибка, и всё это с подсветкой синтаксиса, так же не забудьте вы вести trace в читаемом виде.

**Задача 2.** Написать универсальный обработчик ошибок для PHP7, который будет отлавливать все возможные исключения.

**Работа с файлами.**

**Задание 1.** Написать скрипт, который удаляет заданную номером строку из файла. Общий алгоритм работы скрипта, следующий:

1. Передать в скрипт номер удаляемой строки (можно через форму,

можно через параметр по протоколу GET).

2. Вывести номер удаляемой строки.

3. Открыть исходный файл в режиме чтения.

4. Открыть новый файл в режиме дозаписи.

5. Организовать цикл построчного вывода исходного файла, каждую строку выводить на печать. Если номер текущей строки меньше чем номер удаляемой строки, уменьшенный на 1, то прочитанную строку записать в новый файл. В противном случае продолжить цикл.

6. Закрыть исходный файл.

7. Закрыть новый файл.

8. Удалить исходный файл, новый файл переименовать, присвоив ему

имя исходного файла.

9. Новый файл открыть в режиме чтения и вывести построчно его содер-

жимое на печать, убедиться, что нужная строка удалена.

**Работа с каталогами.**

**Задание 1.** Найдите все файлы из папки 'test', в их содержимом найдите тег

<h1>текст</h1>. Переименуйте все файлы на их h1.

**Задание 2.** Удалите из папки 'test' все файлы размером более 1мб.

**Задание 3.** Имеется папка с файлами, узнайте размер этой папки.

**Задание 4.** Имеется папка с подпапками, узнайте размеры всех подпапок

папки и выведите их на экран.

**Ход работы**

Изучив теоритический материал, преступил к выполнению заданий.

**Листинг 1** – обработка исключений

<?php

// функция обработки ошибок

function myErrorHandler($errno, $errstr, $errfile, $errline)

{

if (!(error\_reporting() & $errno)) {

// Этот код ошибки не включён в error\_reporting,

// так что пусть обрабатываются стандартным обработчиком ошибок PHP

return false;

}

// может потребоваться экранирование $errstr:

$errstr = htmlspecialchars($errstr);

switch ($errno) {

case E\_USER\_ERROR:

echo "<b>Пользовательская ОШИБКА</b> [$errno] $errstr<br />\n";

echo " Фатальная ошибка в строке $errline файла $errfile";

echo ", PHP " . PHP\_VERSION . " (" . PHP\_OS . ")<br />\n";

echo "Завершение работы...<br />\n";

exit(1);

case E\_USER\_WARNING:

echo "<b>Пользовательское ПРЕДУПРЕЖДЕНИЕ</b> [$errno] $errstr<br />\n";

break;

case E\_USER\_NOTICE:

echo "<b>Пользовательское УВЕДОМЛЕНИЕ</b> [$errno] $errstr<br />\n";

break;

default:

echo "Неизвестная ошибка: [$errno] $errstr<br />\n";

break;

}

/\* Не запускаем внутренний обработчик ошибок PHP \*/

return true;

}

// функция для тестирования обработчика ошибок

function scale\_by\_log($vect, $scale)

{

if (!is\_numeric($scale) || $scale <= 0) {

trigger\_error("log(x) для x <= 0 не определён, вы используете: scale = $scale", E\_USER\_ERROR);

}

if (!is\_array($vect)) {

trigger\_error("Некорректный входной вектор, пропущен массив значений", E\_USER\_WARNING);

return null;

}

$temp = array();

foreach($vect as $pos => $value) {

if (!is\_numeric($value)) {

trigger\_error("Значение на позиции $pos не является числом, будет использован 0 (ноль)", E\_USER\_NOTICE);

$value = 0;

}

$temp[$pos] = log($scale) \* $value;

}

return $temp;

}

// переключаемся на пользовательский обработчик

$old\_error\_handler = set\_error\_handler("myErrorHandler");

// вызовем несколько ошибок, во-первых, определим массив с нечисловым элементом

echo "vector a\n";

$a = array(2, 3, "foo", 5.5, 43.3, 21.11);

print\_r($a);

// теперь создадим ещё один массив

echo "----\nvector b - a notice (b = log(PI) \* a)\n";

/\* Значение на позиции $pos не является числом, будет использован 0 (ноль)\*/

$b = scale\_by\_log($a, M\_PI);

print\_r($b);

// проблема, мы передаём строку вместо массива

echo "----\nvector c - a warning\n";

/\* Некорректный входной вектор, пропущен массив значений \*/

$c = scale\_by\_log("not array", 2.3);

var\_dump($c); // NULL

// критическая ошибка, логарифм от неположительного числа не определён

echo "----\nvector d - fatal error\n";

/\* log(x) для x <= 0 не определён, вы используете: scale = $scale \*/

$d = scale\_by\_log($a, -2.5);

var\_dump($d); // До сюда не дойдём никогда

?>

**Листинг 2** – файлы

<body>

<H1> Удаление строки: </H1>

<FORM method="get">

Введите число: <INPUT type="text" name="row">

<P> <INPUT type="submit" value="Результат">

</FORM>

</body>

<?php

if(isset($\_GET["row"]))

{$row = $\_GET["row"];

echo $row ." <br>" ;

$filename = 'co.txt';

$handle = fopen($filename, "r");

$myfile = fopen("testfile.txt", "w");

$file\_arr = file($filename);

$lines = count($file\_arr);

for ($i = 0; $i < $lines; $i++) {

echo htmlspecialchars($file\_arr[$i]), '<br />';

if($row-1 != $i) fwrite($myfile, htmlspecialchars($file\_arr[$i]));

}

echo "<br>";

fclose($handle);

fclose($myfile);

unlink($filename);

rename("testfile.txt", "co.txt" );

$file\_arr = file($filename);

$lines = count($file\_arr);

for ($i = 0; $i < $lines; $i++) {

echo htmlspecialchars($file\_arr[$i]), '<br />';

}

$\_GET["row"] = null;

$row= null;

}

?>

**Листинг 3** – каталоги

<?php

$dir = scandir('../test');

foreach($dir as $elem) {

if(is\_file('test/'.$elem)) {

preg\_match\_all('#\<h1\>(.+)\</h1\>#', file\_get\_contents('test/'.$elem), $arr);

rename('test/'.$elem, 'test/'.$arr[1][0].'.txt');

}

}

?>

<?php

$dir = scandir('../test');

foreach($dir as $elem) {

if(is\_file('../test/'.$elem) and filesize('../test/'.$elem)/1024/1024 > 1) {

unlink('../test/'.$elem);

}

}

?>

<?php

$dir = scandir('../test');

$sum = 0;

foreach($dir as $elem) {

if(is\_file('../test/'.$elem)) {

$sum += round(filesize('../test/'.$elem)/1024/1024);

}

}

echo $sum.' мб';

?>

<?php

$arr = [];

function show\_dir($dir\_name)

{ $sum = 0;

$dir = scandir($dir\_name);

unset($dir[0], $dir[1]);

foreach($dir as $elem) {

$str = $dir\_name;

if(is\_file($dir\_name.'/'.$elem)) {

$sum += round(filesize($dir\_name.'/'.$elem)/1024/1024);

}

if(is\_dir($dir\_name.'/'.$elem)) {

$str .= '/'.$elem;

$arr [$elem]= show\_dir($str);

}

}

return $sum;

}

echo show\_dir('../test');

?>

**Вывод:** ознакомились с теоретическим материалом и базовыми понятиями: пространства имен, типизация данных, а также научились работать со строками и файлами cookie.