Практическое задание № 3 – Файловый ввод/вывод

Цель работы:

Изучение технологии и получение практических навыков работы с библиотекой файлового ввода/вывода в PHP .

**Теоретические сведения**

Как нетрудно предположить, входные и выходные потоки данных интенсивно используются при разработке web-приложений. Не ограничиваясь простым чтением/записью файлов, РНР предоставляет в распоряжение программиста средства просмотра и модификации серверной информации, а также запуска внешних программ. Этим средствам и посвящена настоящая глава.

*Проверка существования и размера файла*

Прежде чем пытаться работать с файлом, желательно убедиться в том, что он существует. Для решения этой задачи обычно используются две функции:

file\_exists( ) и is\_file( ).

file\_exists( )

Функция filе\_ехists ( ) проверяет, существует ли заданный файл. Если файл существует, функция возвращает TRUE, в противном случае возвращается FALSE. Синтаксис функции file\_exists( ):

bool file\_exists(string файл)

Пример проверки существования файла:

if (! file\_exists ($filename)) :

print "File $filename does not exist!";

endif:

is\_file( )

Функция is\_file( ) проверяет существование заданного файла и возможность выполнения с ним операций чтения/записи. В сущности, is\_file( ) представляет собой более надежную версию file\_exists( ), которая проверяет не только факт существования файла, но и то, поддерживает ли он чтение и запись данных:

bool is\_file(string файл)

Следующий пример показывает, как убедиться в существовании файла и возможности выполнения операций с ним:

$file = "somefile.txt";

if (is\_file($file)) :

print "The file $file is valid and exists!";

else :

print "The file $file does not exist or it is not a valid file!";

endif:

Убедившись в том, что нужный файл существует и с ним можно выполнять различные операции чтения/записи, можно переходить к следующему шагу — открытию файла.

filesize( )

Функция filesize( ) возвращает размер (в байтах) файла с заданным именем или FALSE в случае ошибки. Синтаксис функции filesize( ):

int filesize(string имя\_файла)

Предположим, вы хотите определить размер файла pastry.txt. Для получения нужной информации можно воспользоваться функцией filesize( ):

$fs = filesize("pastry.txt"); print "Pastry.txt is $fs bytes.";

Выводится следующий результат:

Pastry.txt is 179 bytes.

Прежде чем выполнять операции с файлом, необходимо открыть его и связать с файловым манипулятором, а после завершения работы с файлом его следует закрыть. Эти темы рассматриваются в следующем разделе.

*Открытие и закрытие файлов*

Прежде чем выполнять операции ввода/вывода с файлом, необходимо открыть его функцией fopen( ).

fopen( )

Функция fopen( ) открывает файл (если он существует) и возвращает целое число — так называемый файловый манипулятор (file handle). Синтаксис функции fopen( ):

int fopen (string файл, string режим [, int включение\_пути])

Открываемый файл может находиться в локальной файловой системе, существовать в виде стандартного потока ввода/вывода или представлять файл в удаленной системе, принимаемой средствами HTTP или FTP.

Параметр файл может задаваться в нескольких формах, перечисленных ниже:

Если параметр содержит имя локального файла, функция fopen( ) открывает этот файл и возвращает манипулятор.

Если параметр задан в виде php://stdin, php://stdout или php://stderr, открывается соответствующий стандартный поток ввода/вывода.

Если параметр начинается с префикса http://, функция открывает подключение HTTP к серверу и возвращает манипулятор для указанного файла.

Если параметр начинается с префикса ftp://, функция открывает подключение FTP к серверу и возвращает манипулятор для указанного файла. В этом случае следует обратить особое внимание на два обстоятельства: если сервер не поддерживает пассивный режим FTP, вызов fopen( ) завершается неудачей. Более того, FTP-файлы открываются либо для чтения, либо для записи.

При работе в пассивном режиме сервер ожидает подключения со стороны клиентов. При работе в активном режиме сервер сам устанавливает соединение с клиентом. По умолчанию обычно используется активный режим.

Ниже приведен пример открытия файла функцией fopen( ). Вызов die( ), используемый в сочетании с fopen( ), обеспечивает вывод сообщения об ошибке в том случае, если открыть файл не удастся:

$file = "userdata.txt"; // Некоторый файл

$fh = fopen($file, "a+") or die("File ($file) does not exist!");

Следующий фрагмент открывает подключение к сайту РНР (http://www.php.net):

$site = "http://www.php.net": // Сервер, доступный через HTTP

$sh = fopen($site., "r");     //Связать манипулятор с индексной страницей Php.net

После завершения работы файл всегда следует закрывать функцией fclose( ).

fclose ( )

Функция fclose( ) закрывает файл с заданным манипулятором. При успешном закрытии возвращается TRUE, при неудаче — FALSE. Синтаксис функции fclose( ):

int fclose(int манипулятор)

Функция fclose( ) успешно закрывает только те файлы, которые были ранее открыты функциями fopen( ) или fsockopen( ). Пример закрытия файла:

$file = "userdata.txt";

if (file\_exists($file)) :

$fh = fopen($file, "r");

// Выполнить операции с файлом

fclose($fh);

else :

print "File Sfile does not exist!";

endif;

*Запись в файл*

С открытыми файлами выполняются две основные операции — чтение и запись.

is\_writeable( )

Функция is\_writeable( ) позволяет убедиться в том, что файл существует и для него разрешена операция записи. Возможность записи проверяется как для файла, так и для каталога. Синтаксис функции is\_writeable( ):

bool is\_writeable (string файл)

Одно важное обстоятельство: скорее всего, РНР будет работать под идентификатором пользователя, используемым web-сервером (как правило, «nobody»). Пример использования is\_writeable( ) приведен в описании функции fwrite( ).

fwrite ( )

Функция fwrite( ) записывает содержимое строковой переменной в файл, заданный файловым манипулятором. Синтаксис функции fwrite( ):

int fwrite(int манипулятор, string переменная [, int длина])

Если при вызове функции передается необязательный параметр длина, запись останавливается либо после записи указанного количества символов, либо при достижении конца строки. Проверка возможности записи в файл продемонстрирована в следующем примере:

<?

// Информация о трафике на пользовательском сайте

$data = "08:13:00|12:37:12|208.247.106.187|Win98";

$filename = "somefile.txt";

// Если файл существует и в него возможна запись

if ( is\_writeable($filename) ) :

// Открыть файл и установить указатель текущей позиции в конец файла

$fh = fopen($filename, "a+");

// Записать содержимое $data в файл

$success - fwrite($fh, $data);

// Закрыть файл

fclose($fh); else :

print "Could not open Sfilename for writing";

endif;

?>

Функция fputs( ) является псевдонимом fwrite( ) и может использоваться всюду, где используется fwrite( ).

fputs( )

Функция fputs( ) является псевдонимом fwrite( ) и имеет точно такой же синтаксис. Синтаксис функции fputs( ):

int fputs(int манипулятор, string переменная [, int длина])

*Чтение из файла*

Несомненно, чтение является самой главной операцией, выполняемой с файлами. Ниже описаны некоторые функции, повышающие эффективность чтения из файла. Синтаксис этих функций практически точно копирует синтаксис аналогичных функций записи.

is\_readable( )

Функция i s\_readable( ) позволяет убедиться в том, что файл существует и для него разрешена операция чтения. Возможность чтения проверяется как для файла, так и для каталога. Синтаксис функции is\_readable( ):

bool is\_readable (string файл]

Скорее всего, РНР будет работать под идентификатором пользователя, используемым web-сервером (как правило, «nobody»), поэтому для того чтобы функция is\_readable( ) возвращала TRUE, чтение из файла должно быть разрешено всем желающим. Следующий пример показывает, как убедиться в том, что файл существует и доступен для чтения:

if ( is\_readable($filename) ) :

// Открыть файл и установить указатель текущей позиции в конец файла

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

else :

print "$filename is not readable!";

endif;

fread( )

Функция fread( ) читает из файла, заданного файловым манипулятором, заданное количество байт. Синтаксис функции fwrite( ):

int fread(int манипулятор, int длина)

Манипулятор должен ссылаться на открытый файл, доступный для чтения (см. описание функции is\_readable( )). Чтение прекращается после прочтения заданного количества байт или при достижении конца файла. Рассмотрим текстовый файл pastry.txt, приведенный в листинге 7.1. Чтение и вывод этого файла в браузере осуществляется следующим фрагментом:

$fh = fopen('pastry.txt', "r") or die("Can't open file!");

$file = fread($fh, filesize($fh));

print $file;

fclose($fh);

Используя функцию fllesize( ) для определения размера pastry.txt в байтах, вы гарантируете, что функция fread( ) прочитает все содержимое файла.

Задание:

На сервере расположить текстовый файл, в котором каждая строка содержит информацию о товаре. Формат строки следующий:

Код\_товара#Наименование\_товара#цена#имя\_файла\_с\_подробным\_описанием

Как видно, разделителем в строке является символ #. Параметр имя\_файла\_с\_подробным\_описанием содержит имя файла, в котором содержится более подробное описание товара. Разработать два документа: в первом будет отображаться список товаров; по нажатию на товар загружается второй документ, отображающий содержимое файла с подробным описанием товара.

# **Содержание отчёта**

Отчёт по практическому заданию должен содержать следующие сведения:

* название и цель работы;
* исходные тексты разработанной программы с комментариями;
* результат работы программы.