Лабораторно упражнение № 6

Работа с файлове

1. Функции за операции с файлове.

PHP предоставя набор от функции за работа с файлове - http://php.saparev.com/ref.filesystem.html.

1.1. fopen() - функция за отваряне на файл. Отваря файл или URL.

Синтаксис:

```
resource fopen (string $filename , string $mode [, bool
$use_include_path [, resource $context ]])
```

Функцията fopen() свързва именуван ресурс, определен чрез параметър filename с поток. При успех връща файлов манипулатор, при грешка **FALSE**.

Примери:

```
$handle = fopen("c:\\data\\info.txt", "r");
$handle = fopen("/home/rasmus/file.txt", "r");
$handle = fopen("/home/rasmus/file.gif", "wb");
$handle = fopen("http://www.example.com/", "r");
$handle =
fopen("ftp://user:password@example.com/somefile.txt",
"w");
```

Под Windows трябва да дублирате всички обратни наклонени, използвани в пътя до файла или трябва да използвате прави наклонени черти.

Параметърът **mode** определя типа на достъп, който искате за потока. Той може да бъде един от следните:

- r - само за четене. Установява указателя на файла в неговото начало.
- r+ - за четене и запис. Установява указателя на файла в неговото начало.
- w - само за запис. Установява указателя на файла в неговото начало и нулира дължината на файла. Ако файлът не съществува се опитва да го създаде.
- w+ - за четене и запис. Установява указателя на файла в неговото начало и нулира дължината на файла. Ако файлът не съществува се опитва да го създаде.
- a - само за запис. Установява указателя на файла в неговия край. Ако файлът не съществува се опитва да го създаде.
- a+ - за четене и запис. Установява указателя на файла в неговия край. Ако файлът не съществува се опитва да го създаде.
- x - само за запис. Установява указателя на файла в неговото начало. Ако файлът съществува, **fopen()** ще върне **FALSE** и ще генерира грешка от тип **E_WARNING**. Ако файлът не съществува се опитва да го създаде.
- x+ - за четене и запис. Установява указателя на файла в неговото начало. Ако файлът съществува, **fopen()** ще върне **FALSE** и ще генерира грешка от тип **E_WARNING**. Ако файлът не съществува се опитва да го създаде.

Забележка: Освен дадените по-горе символи, атрибутът mode може да съдържа и символа *'b'* – *bynary*, при отварянето на файлове с **fopen().** От съображение за съвместимост се препоръчва винаги да се използва 'b' при отварянето на файлове.

```
$fp = fopen('data.txt', 'wb');
```

1.2. fgets() – функция за прочитане на ред от текстов файл. Функцията връща string, като чете до зададена дължина или чете до края на файл, ако не е открит символ за край на ред.

Синтаксис:

```
string fgets (resource $handle [, int $length ])
```

където

`handle` - Файлов указател, който трябва да бъде валиден и да сочи към файл, който е бил отворен успешно чрез функция fopen().

`length` - на дължина

1.3. feof() - Проверява дали указателят е в края на файла. Връща TRUE, когато файловият указател е в края на файла (EOF) или се е получила грешка (включително и изтичане на времето за изчакване (timeout)). В останалите случаи връща FALSE.

Синтаксис:

```
bool feof (resource $handle)
```

`handle` - файлов указател, който трябва да бъде валиден и да сочи към файл, който е бил отворен успешно чрез функция fopen(). Ако подадения файлов указател е невалиден може да се получи безкраен цикъл, защото feof() не може да върне TRUE.

Пример 1. Пример за прочитане на файл, ред по ред. Използват се функции fgets() - за прочитане на ред от файла и за край на файл - feof().

```
<pre>
<?php
    $fp = @fopen("data.txt", "r");
    if ($fp) {
        while (!feof($fp)) {
            //$buffer = fgets($fp, 4096);
            $buffer = fgets($fp);
            echo $buffer;
        }
        fclose($fp);
    }
?>
</pre>
```

Или

```
<?php
    echo "<pre>";
    $fp = fopen("proba.txt", "r") or die("<br>can't open the
    file for reading");
    //$fp = fopen("proba.txt", "r") or exit("<br>can't open
    the file for reading");
        while (!feof($fp)) {
            //$buffer = fgets($fp, 4096);
            $buffer = fgets($fp);
            echo $buffer;
        }
        fclose($fp);
```

```
    echo "</pre>";
?>
```

В примера се прочита и извежда последователно по редове съдържанието на файла, който трябва да е в същата папка в която е php файла. Етикетът за преформатиране "<pre>" е необходим за да отработи символа за преминаване на нов ред при извеждане на текста във Web браузъра.

1.4. fgetc() - прочита един символ от текущата позиция на файловия указател.

Синтаксис:

```
string fgetc (resource $handle)
```

където

`handle` - Файлов указател, който трябва да бъде валиден и да сочи към файл, който е бил отворен успешно чрез функция fopen().

Тази функция може да върне булева стойност FALSE, но може също да върне небулева стойност, която се интерпретира като FALSE, като 0 или "". Използвайте оператора === или !== за проверка на връщаната стойност от тази функция.

Пример 2. Пример за прочитане на файл, символ по символ.

```
<?php
    echo "<pre>";
    $fp = fopen("data.txt", "r") or exit("<br>can't open the
    file for reading");
        while (!feof($fp)) {
            $buffer = fgetc($fp);
            echo $buffer;
        }
        fclose($fp);
    echo "</pre>";
?>
```

или

```
<?php
    echo "<pre>";
    $fp = fopen("data.txt", "r") or die("<br>can't open the
    file for reading");
    $char = fgetc($fp);
    while (false !== $char)
        { echo "$char"; $char = fgetc($fp); }
    echo "</pre>";
?>
```

1.5. fread() – прочита указан брой символи, започвайки от файловия указател, указан чрез handle . Четенето спира при случването на едно от следните събития:

- когато са прочетени указания брой символи;
- когато се достигне до края на файла;
- когато пакет е станал наличен (за мрежови потоци);

- 8192 байта са били изчетени (след отваряне на потока);

Синтаксис:

```
string fread (resource $handle , int $length)
```

където

handle - Указателен файлов ресурс, който обикновено се създава посредством fopen().

length - Брой символи, които трябва да бъдат прочетени.

Пример 3. Прочитане на файл чрез използването на fread().

```
<?php
    // изчита съдържанието на файл в низ
    $filename = "data.txt";
    $handle = fopen($filename, "r");
    $contents = fread($handle, filesize($filename));
    print_r($contents); //отпечатва прочетения низ в прозореца
    на браузъра
    fclose($handle);
?>
```

!!!При операционни системи, където има разлика между двоични и текстови файлове (примерно Windows), файлът трябва да бъде отворен с 'b' режим при извикването на fopen(). Ако е необходимо да извлечете съдържанието на файл в низ - използвайте функция **file_get_contents()**, която има по-добра производителност спрямо горния код. Функцията връща резултата в низ, като започва четенето от отместване offset и изчитайки maxlen байта. При грешка file_get_contents() ще върне FALSE.

```
string file_get_contents (string $filename [, int $flags [, resource
$context [, int $offset [, int $maxlen ]]]])
```

Забележка: Функцията fread() чете от текущата позиция на указателя за файла. Използвайте ftell(), за да разберете текущата позиция и rewind(), за да върнете указателя в началото на файла.

1.6. fwrite() - Функцията записва съдържанието на string във файловия поток сочен от указателя handle . Функцията връща броя на записаните байтове или FALSE при грешка.

Синтаксис:

```
int fwrite (resource $handle , string $string [, int $length ])
```

където

handle - Указателен файлов ресурс, който обикновено се създава посредством fopen().

string - Низът, който ще бъде записан.

length - Ако е подаден параметър length , записа ще спре след като length байта са записани или докато се достигне края на string , което от двете се случи първо.

Забележка: При операционни системи, които правят разлика между двоични и текстови файлове (като Windows), файлът трябва да е отворен с флаг "b" при извикването на fopen(). Ако запишете два пъти в указател към файл, вторите данни ще бъдат добавени към края на файла, т.е. следващия пример няма да работи както е очаквано.

Пример 4. Записване на данни във файл чрез използването на fwrite().

```
<?php
     $fp = fopen('data.txt', 'w');
     fwrite($fp, '1');
     fwrite($fp, '23');
     fclose($fp);
     // Съдържанието на 'data.txt' сега е 123, а не 23!
?>
```

Пример 5. Записване на данни във файл чрез използването на fwrite() и file_put_contents().

```
<html>
<body>
<?php
     $myFile = "data.txt";
     $fh = fopen($myFile, 'w') or die("can't open file");
     $stringData = "Иван Иванов\n";
     fwrite($fh, $stringData);
     $stringData = "Георги Тодоров\n";
     fwrite($fh, $stringData);
     $s="Здравейте!";
     file_put_contents($myFile,$s,FILE_APPEND);//запис във
     файла, FILE_APPEND
     fclose($fh);
     //Изтрива файла
     //unlink($myFile);?>
</body>
</html>
```

Резултат:

```
Иван Иванов
Георги Тодоров
Здравейте!
```

1.7. **unlink ()** – Функцията изтрива файл.
     `unlink($myFile)` - изтрива файла от Пример 5.

1.8. **fputs()** - Псевдоним на функцията fwrite()

1.9. **file_put_contents()** – Функцията записва низ във файл.

Синтаксис:

```
int file_put_contents (string $filename , mixed $data [, int $flags [,
resource $context ]])
```

където

filename - Път до файла, в който ще бъдат записвани данните.

data - Данните, който ще бъдат записани. Могат да са string, array или stream. Ако data е от тип stream, остатъкът в буфера на потока ще бъде записан в посочения файл. Това е подобно на използването на stream_copy_to_stream().
Също така, може да използвате параметъра data като едномерен масив. Това е равнозначно на file_put_contents($filename, implode('', $array)).

flags - Стойността на flags може да бъде комбинация от следните:

FILE_USE_INCLUDE_PATH	Търси файла *filename* и в пътищата за включване.
FILE_APPEND	Ако файлът *filename* съществува, данните ще бъдат добавени към него вместо да бъдат презаписани отгоре.
LOCK_EX	Придобива изключително заключване върху файла докато се записва в него.
FILE_TEXT	Данните от *data* се записват в текстов режим. Ако се използва unicode семантика, кодирането по подразбиране ще е UTF-8. Можете да използвате различно кодиране чрез създаването на context или чрез използването на **stream_default_encoding()**, за да промените подразбиращото се кодиране. Този флаг не може да бъде използван едновременно с **FILE_BINARY**. Този флаг е наличен от PHP6.
FILE_BINARY	Данните от *data* се записват в двоичен режим. Това е режимът използван по подразбиране. Не може да бъде използван едновременно с **FILE_TEXT**. Този флаг е наличен от PHP6.

Функцията извършва същото, както и последователността fopen(), fwrite() и fclose() (извикани успешно). Ако файлът filename не съществува ще бъде създаден. Ако съществува ще бъде презаписан, освен ако не е подаден флаг FILE_APPEND.

Добавете към Пример 5. и тествайте:

```
...
$s="Лабораторно упражнение по Web приложения!";
file_put_contents('data.txt',$s);//запис във файла
```

1.10. file_exists() – Функцията проверява дали даден файл съществува. Функцията връща като резултат TRUE ако файлът или директорията подадени с параметър filename съществуват, ако не - връща FALSE

Синтаксис:

```
bool file_exists (string $filename)
```

където

filename - Път до файла или директорията.

Пример 6. Използване на функцията file_exists().

```
<?php
$filename = '/path/to/foo.txt';
if (file_exists($filename))
    echo "Файлът $filename съществува.";
else
    echo "Файлът $filename не съществува.";
?>
```

Резултат:

Файлът /path/to/foo.txt не съществува.

1.11. filesize() - Функция за проверка на размера на файл. Връща като резултат размера на файл в байтове или FALSE при грешка.

Синтаксис:

```
int filesize (string $filename)
```

където

filename - Името на файла.

Пример 7. Използване на функции filesize() и file_put_contents().

```
<?php
```

```
    $fp = fopen('data.txt', 'wb');
    $s="Здравейте!";
    file_put_contents('data.txt',$s);
    fclose($fp);
    $filename = 'data.txt';
    echo $filename . ': ' . filesize($filename) . ' bytes';
    //data.txt: 19 bytes
?>
```

1.12. is_readable() – Функция, която проверява дали даден файла е разрешен за четене. Връща TRUE ако файлът или директорията определена от filename съществува и може да се чете. В противен случай връща FALSE.

Синтаксис:

```
bool is_readable(string $filename)
```

където

filename - Името на файла, който ще бъде проверен.

1.13. is_writable() – Функция, която проверява дали файла е разрешен за запис. Връща TRUE ако filename съществува и може да се записва в него. Параметърът filename може да бъде и директория, като по този начин се проверява дали може да се записва в директорията.

Синтаксис:

```
bool is_writable(string $filename)
```

където

filename - Името на файла, който ще бъде проверен.

1.14. is_executable() - Функция, която проверява дали файла е изпълним. Връща TRUE ако файла filename е изпълним или FALSE ако не е.

Синтаксис:

```
bool is_executable(string $filename)
```

където

filename - Името на файла, който ще бъде проверен.

Връща TRUE ако файла filename е изпълним или FALSE ако не е.

Пример 8. Използване на функции is_readable(), is_writable() и is_executable().

```
<?php
    $filename = 'data.txt';
    if (is_readable($filename)) {
        echo 'Файлът може да се чете.';
    } else {
        echo 'Файлът НЕ може да се чете.';
    }
    echo "<br>";
    if (is_writable($filename))
        echo 'Във файла може да се записва.';
    else
        echo 'Във файла НЕ може да се записва.';
    echo "<br>";
    if (is_executable($filename))
        echo $filename.' е изпълним.';
    else
        echo $filename.' не е изпълним.';
?>
```

Резултат:

Файлът може да се чете.
Във файла може да се записва.
data.txt не е изпълним.

1.15. rmdir() - Функция за премахване на директория. Опитва се да изтрие директория с посоченото име. Директорията трябва да е празна и да имате необходимите права за изтриване. Функцията Връща TRUE при успех или FALSE при неуспех.

Синтаксис:

```
bool rmdir(string $dirname [, resource $context ])
```

1.16. mkdir() - Функция за създаване на директория. Опитва да създаде директория с име зададено чрез pathname .

Синтаксис:

bool mkdir (string $pathname [, int $mode [, bool $recursive [, resource $context]]])

където

pathname – Път до директория

mode - Режимът по подразбиране е 0777, което е равносилно на възможно най-неограничения достъп. Когато искате да подадете и параметър за режим (mode) трябва да го зададете в осмичен формат, т.е. с водеща нула отпред.

recursive - Стойността му по подразбиране е FALSE.

context - Контекстната поддръжка е добавена от PHP 5.0.0.

Пример 9.

```
<?php
	mkdir("dir", 0700);
	mkdir("folders");
	mkdir("folder");
	rmdir("folder");
?>
```