Web2, curs09, Schita notiuni

Lucrul cu fisiere in PHP.

1. **Accesarea fisierelor**
   - deschiderea unui fisier: ***fopen($filename, $mode)***

Posibilitati pentru **$mode**	
'r'	Fisierul este deschis doar pentru citire. Pointerul este la inceputul fisierului
'r+'	Fisierul este deschis pentru citire si scriere. Pointerul este la inceputul fisierului
'w'	Fisierul este deschis doar pentru scriere. Daca fisierul nu exista, el este creat. Daca exista, se sterge tot din fisier, pointerul fiind plasat la inceput
'w+'	Ca 'w', insa fisierul este deschis pentru citire si pentru scriere
'a'	Deschis pentru scriere. Daca fisierul nu exista, el este creat. Pointerul este plasat la sfarsit.
'a+'	La fel ca 'a' dar este deschis atat pentru scriere cat si pentru citire.
'x'	Creeaza si deschide un fisier pentru scriere. Daca fisierul exista, functia returneaza FALSE, si genereaza un Warning
'x+'	La fel ca 'x' insa deschide fisierul si pentru citire

Upload. Mutarea fisierelor intr-un folder

6 variabile din php.ini care pot influenta uploadul de fisiere prin metoda POST

1. "*file_uploads*" – variabila booleana care indica daca este permis sau nu uploadul de fisiere
2. *"max_execution_time"*

- determina numarul maxim de secunde in care un script poate rula inainte de a fi intrerupt automat. Se va creste valoarea acestei variabile daca se anticipeaza upload de fisiere mari sau viteza mica la transfer.

3. *"max_input_time"* - controleaza timpul maxim permis pentru preluarea datelor de intrare, inclusiv uploadul de fisiere prin POST. Se va creste valoarea acestei variabile daca se anticipeaza upload de fisiere mari sau viteza mica la transfer.
4. *"upload_max_filesize"* - determina dimensiunea maxima acceptata pentru un fisier la upload; are prioritate fata de campul ascuns MAX_FILE_SIZE
5. *"post_max_size"* - determina dimensiunea maxima a datelor care pot fi transmise printr-o cerere POST (inclusiv upload de fisiere); trebuie sa fie cel putin cat upload_max_filesize
6. *"upload_tmp_dir"* - determina directorul temporar folosit pentru upload-ul de fisiere

Web2, curs07, Schita notiuni

II. PHP – lucrul cu fisiere si directoare

1. **Upload de fisiere**

aplicatia 06_ex02

2. **Operatii cu directoare**

a) mkdir()

- Prototip: *bool mkdir (string $pathname [, int $mode= 0777 [, bool $recursive= false]])*
  - Creeaza un director in locatia specificata de $pathname. $pathname este calea catre directorul ce trebuie creat.
  - Daca $pathname cuprinde mai multe nivele de directoare ce nu exista, trebuie sa folosesc al treilea parametru avand valoarea **true** ce indica crearea tuturor directoarelor din $pathname.
  - **Al doilea parametru, este un numar in octal (numerele in baza 8 in php trebuie scrise cu 0 in fata), ce reprezinta permisiunile pentru acel director. Acest parametru, $mode, este ignorat in Windows.**

b) rmdir()

- Prototip: *bool rmdir (string $dirname)*
  - Sterge directorul $dirname, daca scriptul are permisiunile necesare si daca directorul este gol. Pentru a sterge un director care nu e gol, trebuie realizata o functie utilizator, ce sterge in mod recursiv subdirectoarele si fisierele din directorul $dirname, si in final sa stearga si $dirname.

c) **opendir($dir)**

- deschide directorul $dir si returneaza o variabila de tip resursa, $d , cu ajutorul careia citim directorul

d) **readdir($d)**

- citeste o intrare in lista directorului reprezentat de $d, si returneaza numele intrarii respective. O intrare poate fi un fisier, un director, sau fisierele speciale "." si ".." ce reprezinta directorul curent respectiv directorul parinte. Daca a ajuns la sfarsit, readdir() va returna FALSE, si atunci stim ca am terminat de listat continutul directorului. Insa intotdeauna trebuie sa comparam valoarea returnata de readdir() din punct de vedere identic cu FALSE, sa ne asiguram ca returneaza intr-adevar FALSE si nu o valoare ce se converteste automat la FALSE cum ar fi 0 (in caz ca un fisier se numeste 0).

- instructiunea $f = readdir($d) va atribui rezultatul lui readdir($d), deci numele fisierului curent, variabilei $f. Rezultatul acestei instructiuni este bineinteles valoarea atribuita, care trebuie sa fie diferita (!==) de FALSE.

2. **Deschiderea, inchiderea, scrierea si cititrea fisierelor**

a) **fopen($filename, $mode)**

- deschide sau creeaza fisierul $filename
- Posibilitati pentru **$mode** – a se vedea tabelul de mai sus

- **$filename** poate fi :
  - O cale relativa la scriptul php. De exemplu: fisiere/a.txt
  - O cale absoluta catre un fisier
  - un url. Ex: http://www.exemplu.com/a.txt. In acest caz, setarea *allow_url_fopen* din php.ini trebuie sa fie On. Aceasta setare se poate schimba conform setarii PHP_INI_SYSTEM , adica doar de administratorul serverului in php.ini sau in httpd.conf
- **Valoarea returnata** de fopen() este FALSE in cazul in care apare o eroare in deschiderea fisierului specificat sau o valoare de tip resursa in caz de succes. Aceasta variabila de tip resursa (denumita deseori $fp sau $handle) este folosita ulterior de functii precum fread() sau fwrite() pentru citirea sau scrierea in fisier.

b) fwrite()

- Prototip: ***int fwrite (resource $handle , string $string [, int $length])***
  - **$handle** - este o variabila de tip resursa, returnata de fopen()
  - **$string** - sirul de caractere (sau bytes) scris in fisier
  - **$length** - argument optional. Daca este furnizat, scrierea in fisier se opreste cand apare prima data unul din cele doua evenimente: se termina string-ul sau cand string-ul scris ajunge la lungimea $length
  - **returneaza** numarul de bytes scrisi in fisier

c) fcloase($handle)

- inchide fisierul

d) file_put_contents()

- Prototip: ***int file_put_contents (string $filename , mixed $data [, int $flags= 0 [, resource $context]])***
  - Functia **file_put_contents()** simplifica scrierea in fisiere, apelarea sa fiind echivalenta cu a apela fopen(), fwrite() si apoi fclose().
  - Primul parametru este numele fisierului ce va fi deschis pentru scriere, apoi urmeaza informatia ce o scriem in fisier, iar al treilea parametru $flags, consta intr-o constanta sau combinatie de constante ce modifica optiunile functiei.
  - Folosind constanta FILE_APPEND , determinam adaugarea textului la sfarsitul fisierului la fiecare apelare a functiei si nu golirea (trunchierea) lui inainte de scriere asa cum ar fi fost implicit.

e) **file_get_contents($filename)** este cea mai simpla functie pentru citirea intregului continut al unui fisier intr-un string. ex: $s=file_get_contents('date.txt'); echo $s;

f) **readfile($filename)** va citi fisierul $filename, si ii va afisa continutul (trimite catre browser).

g) **fread($fp, $length)** este folosita pentru a citi $length bytes dintr-un fisier si returneaza FALSE in caz de eroare sau string-ul citit.

h) **file($filename)** este foarte utila, permite citirea intregului continut al unui fisier, si returnarea lui intr-un array. Fiecare element al array-ului este o linie din fisierul respectiv.