Pagini WEB cu PHP 4

Mihai Scorțaru, Claudiu Soroiu

Reluăm seria de articole dedicate limbajului de interpretare PHP ajuns acum la versiunea 5. În cadrul acestui articol vom prezenta cele mai importante funcții de prelucrare a fișierelor și directoarelor puse la dispoziție de acesta.

Manipularea fișierelor

Limbajul *PHP* oferă un set de funcții pentru a facilita accesul la conținutul fișierelor. Sintaxa majorității dintre acestea este similară funcțiilor de manipulare a fișierelor prezente în limbajul *C/C++*. Datorită faptului că limbajul *PHP* este independent de platformă, printre numele funcțiilor se regăsesc și nume cunoscute pe platformele de operare **nix* și *Microsoft Windows*.

Funcția fopen

Această funcție se folosește pentru deschiderea unui fișier în vederea citirii sau scrierii datelor din/în acesta.

Funcția are patru parametri, dintre care ultimii doi sunt opționali.

Primul parametru îl reprezintă numele fișierului care urmează a fi deschis. Fișierul poate fi și unul aflat la distanță. Un fișier este recunoscut de *PHP* a fi la distanță dacă este accesibil printr-unul dintre protocoalele ***HTTP*** (***H**yper **T**ext **T**ransfer **P**rotocol*), ***HTTPS*** (***S**ecure HTTP*), ***FTP*** (***F**ile **T**ransfer **P**rotocol*) sau ***FTPS*** (***S**ecure FTP*).

Fișierele standard de intrare și ieșire pot fi accesate folosind următorii identificatori:

- *php://stdin* — fișierul standard de intrare pentru procesul curent creat de *PHP*;
- *php://stdout* — fișierul standard de ieșire pentru procesul curent creat de *PHP*;
- *php://stderr* — fișierul standard de erori pentru procesul curent creat de *PHP*;
- *php://input* — cu ajutorul acestui identificator se poate accesa conținutul datelor transmise paginii prin intermediul metodei *POST*;
- *php://output* — cu ajutorul acestui identificator se pot scrie date direct în pagina web.

Fișierele *php://input* și *php://stdin* pot fi deschise numai în scopul citirii, în timp ce fișierele *php://output*, *php://stdout* și *php://stderr* pot fi deschise numai pentru scriere.

Scrierea datelor în fișierul *php://output* are același efect ca și utilizarea funcțiilor **`print`** și **`echo`**.

Cel de-al doilea parametru este de tipul șir de caractere și reprezintă modul de deschidere a fișierului care poate fi:

- `"r"` — deschide fișierul pentru citire și poziționează *pointer*-ul la începutul acestuia;
- `"r+"` — deschide fișierul pentru citire și scriere și poziționează *pointer*-ul la începutul acestuia;
- `"w"` — deschide fișierul pentru scriere, poziționează *pointer*-ul la începutul acestuia și șterge conținutul; dacă fișierul nu există, acesta este creat.
- `"w+"` — deschide fișierul pentru citire și scriere, poziționează *pointer*-ul la începutul acestuia și șterge conținutul; dacă fișierul nu există, acesta este creat.
- `"a"` — deschide fișierul pentru scriere și poziționează *pointer*-ul la sfârșitul acestuia, însă fără a șterge conținutul; dacă fișierul nu există, acesta este creat.
- `"a+"` — deschide fișierul pentru citire și scriere și poziționează *pointer*-ul la sfârșitul acestuia, însă fără a șterge conținutul; dacă fișierul nu există, acesta este creat.

La sfâșitul șirului care reprezintă modul de deschidere a fișierului mai poate apărea și caracterul `"b"`. Acesta este util atunci când se dorește deschiderea fișierului în mod binar și nu în mod text. Pe sistemele de operare **nix* nu are nici un efect, în schimb pentru portabilitate este indicată folosirea lui.

Cel de-al treilea parametru este de tip întreg și poate avea valorile `0` sau `1` și indică funcției faptul că se poate folosi (valoarea 1) sau nu (valoarea `0`) lista de directoare, cu care interpretorul a fost configurat, pentru a găsi fișierul care se dorește a fi deschis, în cazul în care numele acestuia nu include și o cale absolută.

Cel de-al patrulea parametru este de tipul `resource` și este folosit în scopul optimizării unor operații.

Funcția `fopen` returnează un număr întreg care reprezintă identificatorul de acces la fișierul care tocmai a fost deschis. Acest identificator va fi transmis tuturor funcțiilor pe care le veți utiliza pentru a accesa date din fișier.

În cazul în care deschiderea fișierului nu poate avea loc din diferite motive, atunci funția returnează valoarea logică **`FALSE`**.

De exemplu, pentru a deschide fișierul cu numele `"test.txt"` pentru a citi date, vom folosi următoarea linie de cod sursă: `$f = fopen("test.txt","r");`. Dacă dorim să scriem date în fișier, atunci îl putem deschide folosind următorul cod: `$f = fopen("test.txt","w");`.

Funcția tmpfile

Această funcție realizează crearea și deschiderea unui fișier temporar pentru scriere și citire. Funcția `tmpfile` nu are parametri și returnează un număr întreg care reprezintă identificatorul de acces al fișierului temporar.

Fișierul temporar va fi șters în mod automat în momentul în care este închis sau când *script*-ul își încheie execuția.

Apelul acestei funcții este identic cu cel al funcției **`fopen`** cu parametrii: un nume de fișier temporar (de obicei generat aleator) și `"w+b"` ca mod de deschidere a fișierului.

Funcția fclose

Această funcție realizează închiderea unui fișier care a fost deschis cu ajutorul funcțiilor `fopen` sau `tmpfile`.

Funcția `fclose` are un singur parametru de tip întreg, care reprezintă identificatorul de acces la fișierul care se dorește a fi închis.

Funcția returnează valoarea logică **`TRUE`** dacă închiderea fișierului a reușit și valoarea logică **`FALSE`** în caz contrar.

Un exemplu de apel al acestei funcții este `fclose($fd)`, unde `fd` reprezintă identificatorul de acces la fișierul care se dorește a fi închis.

Funcția feof

Această funcție verifică dacă s-a ajuns la sfârșitul fișierului sau nu.

Funcția `feof` are un singur parametru de tip întreg, care reprezintă identificatorul de acces la fișierul care se dorește a fi închis.

Funcția returnează valoarea logică **`TRUE`** dacă am ajuns la sfârșitul fișierului și valoarea logică **`FALSE`** în caz contrar.

Funcțiile fscanf

Această funcție citește date dintr-un fișier text și are doi parametri obligatorii.

Primul parametru este de tip întreg și reprezintă identificatorul de acces la fișierul din care se face citirea, iar cel de-al doilea parametru este de tipul șir de caractere și indică funcției formatul intrării. Formatul acestuia din urmă este similar formatului folosit pentru funcția `fscanf` prezentă în limbajul *C/C++*.

Ceilalți parametri reprezintă referințele la variabilele care vor conține rezultatul citirii.

Funcția returnează un vector cu valorile citite, deci prezența variabilelor nu mai este obligatorie ca în cazul limbajului *C/C++*.

De exemplu, dacă din fișierul `"c:\test.txt"` dorim să citim un număr și un șir de caractere separate printr-un spațiu, atunci vom folosi următorul cod:

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu fscanf
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $f = fopen("c:\\test.txt","r");
      $rez = fscanf($f,"%d %s\n");
      list($nr,$sir) = $rez;
      fclose($f);
      echo "Am citit numarul: ".$nr."<BR>";
      echo "si sirul: ".$sir;
    ?>
  </BODY>
</HTML>
```

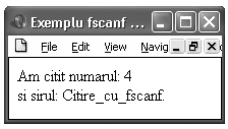

Figura 1: Funcția fscanf

În figura 1 poate fi observat efectul execuției acestui cod.

Funcția fread

Această funcție se folosește pentru citirea datelor din fișiere binare. Pentru compatibilitate cu toate sistemele de operare se recomandă ca la deschiderea fișierului, pentru citire, să se adauge caracterul `"b"` la parametrul care reprezintă modul de deschidere a fișierului.

Funcția `fread` are doi parametri obligatorii. Primul parametru este de tip întreg și reprezintă identificatorul de acces al fișierului din care se dorește să se citească date, iar al doilea parametru, tot de tip întreg, reprezintă numărul de octeți care se doresc a fi citiți.

Rezultatul returnat de această funcție este de tip șir de caractere care este format din octeții care s-au citit din fișierul de intrare.

În cazul în care în fișierul din care se citesc date nu mai este disponibil numărul de octeți specificat în cadrul celui de-al doilea parametru, atunci funcția returnează toți octeții disponibili de la poziția curentă din fișier și până la sfârșitul acestuia.

În continuare vă prezentăm un exemplu de folosire a acestei funcții, iar în figura 2 poate fi observat efectul execuției codului.

Codul este compatibil cu toate sistemele de operare pe care limbajul *PHP* poate fi instalat.

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu fread
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $fis = "c:\\test.txt";
      $fd = fopen($fisr,"rb");
      $continut = fread($fd,filesize($fis));
      fclose($fd);
      echo "Continutul fisierului este: ";
      echo $continut;
    ?>
  </BODY>
</HTML>
```

Figura 2: Funcția fread

Funcția fgetc

Această funcție citește următorul caracter disponibil din fișierul de intrare.

Funcția are un singur parametru de tip întreg care reprezintă identificatorul de acces la fișierul din care se dorește să se facă citirea.

Funcția `fgetc` returnează un șir de caractere, în cazul în care nu s-a ajuns la sfârșitul fișierului, format dintr-un singur caracter, și anume, cel citit din fișier.

În cazul în care ne aflăm la sfârșitul fișierului, funcția `fgetc` returnează valoarea logică **`FALSE`**.

În continuare vă prezentăm un exemplu de utilizare a acestei funcții și în figura 3 poate fi observat efectul execuției acestui cod:

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu fgetc
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $fisier = "c:\\test.txt";
      $fd = fopen($fisier,"r");
      $car = fgetc($fd);
      fclose($fd);
      echo "Primul caracter din fisier este: ";
      echo $car;
    ?>
  </BODY>
</HTML>
```

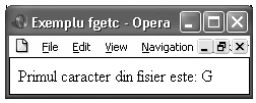

Figura 3: Funcția fgetc

Funcția fgets

Această funcție se folosțe pentru a citi o linie dintr-un fișier de tip text. Sintaxa acestei funcții este similară cu cea a funcției `fread`, cu mențiunea că cel de-al doilea parametru este opțional.

Funcția `fgets` returnează un șir de caractere care conține datele citite. Citirea datelor din fișier se încheie în momentul în care s-a citit un număr de caractere egal cu valoarea celui de-al doilea parametru (în cazul în care acesta este prezent), în momentul întâlnirii unui sfârșit de linie sau în momentul în care s-a ajuns la sfârșitul fișierului.

În cazul în care ne aflăm la sfârșitul fișierului, funcția `fgets` returnează valoarea logică **`FALSE`**.

În continuare vom prezenta un exemplu de utilizare a acestei funcții:

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu fgets
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $fisier = "c:\\test.txt";
      $fd = fopen($fisier,"r");
      $linie = fgets($fd);
      fclose($fd);
      echo "Prima linie din fisier este: ";
      echo $linie;
    ?>
  </BODY>
</HTML>
```

Figura 4: Funcția fgets

În acest exemplu am omis cel de-al doilea parametru. În figura 4 poate fi observat efectul execuției acestui cod.

Funcția fgetss

Această funcție citește o linie dintr-un fișier *HTML*. Spre deosebire de funcția **`fgets`**, funcția **`fgetss`** are trei parametri.

Primii doi parametri sunt identici cu cei folosiți în cazul funcției **`fgets`**. Cel de-al treilea parametru este opțional, este de tip șir de caractere și reprezintă o listă de marcaje *HTML* sau *PHP* pe care funcția nu trebuie să le elimine în momentul citirii.

Această funcție returnează un șir de caractere care reprezintă datele din fișierul *HTML* care au fost citite.

Citirea datelor se încheie în momentul în care s-au citit atâtea caractere câte indică cel de-al doilea parametru sau în momentul în care am ajuns la sfârșitul fișierului.

În continuare prezentăm un exemplu de utilizare a acestei funcții și în figura 6 poate fi observat efectul execuției acestui cod pentru fișierul din figura 5.

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu fgetss
    </TITLE>
  </HEAD>
```

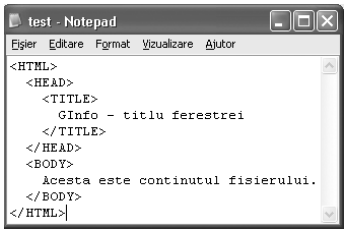

Figura 5: Fișier HTML

```
<BODY>
  <?PHP
    $fis = "c:\\test.html";
    $fd = fopen($fis,"r");
    echo "Continutul fisierului HMTL este:";
    while (!$l = fgetss($fd,filesize($fis))){
      echo $l."<BR>";
    }
  fclose($fd);
?>
</BODY>
</HTML>
```

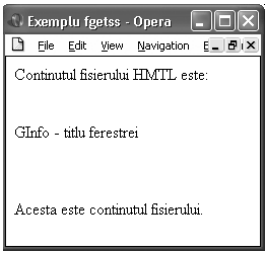

Figura 6: Funcția fgetss

Se observă că rezultatul obținut după citirea tuturor liniilor unui fișier *HTML* conține foarte multe rânduri goale. Acest fapt se datorează liniilor care conțin doar marcaje *HTML* și, prin eliminarea lor, șirul returnat devine vid.

Funcția fgetcsv

Această funcție citește date dintr-un fișier ***CSV*** (***C**omma **S**eparated **V**alues - valori separate prin virgulă*).

Un fișier de tip *CSV* are formatul text și conține mai multe înregistrări. Este o reprezentare textuală a unui fișier cu tip. Pe fiecare linie a unui astfel de fișier se află mai multe valori ale unor câmpuri care sunt delimitate printr-un separator. De obicei acest separator este caracterul "," (virgula), de unde și numele acestui tip de fișier.

Funcția `fgetcsv` are trei parametri dintre care primii doi sunt obligatorii. Primul parametru este de tip întreg și reprezintă identificatorul de acces la fișierul de intrare, al doilea parametru este de tip întreg și reprezintă numărul maxim de octeți care se vor procesa la o citire și trebuie să aibă o valoare mai mare decât lungimea celei mai mari linii din fișierul *CSV*, iar al treilea parametru este de tip șir de caractere care trebuie să aibă lungimea 1 și reprezintă separatorul care se va folosi la procesarea datelor.

Funcția returnează, în cazul în care nu ne aflăm la sfârșitul fișierului de intrare, un tablou de șiruri de caractere care reprezintă valorile citite.

În cazul în care ne aflăm la sfârșitul fișierului, funcția `fgetcsv` returnează valoarea logică **`FALSE`**.

În continuare prezentăm un exemplu de utilizare a acestei funcții pentru un fișier *CVS* în care fiecare linie nu are o lungime mai mare de 1000 de caractere:

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu fgetcsv
    </TITLE>
  </HEAD>
    <?PHP
      $lin = 1;
      $fp = fopen("test.csv","r");
      while ($date = fgetcsv($fp, 1000, ",")) {
        $num = count($date);
        echo "<P> $num câmpuri pe linia
                                    $lin: <BR>";
        $lin++;
        for($c=0; $c < $num; $c++){
          echo $date[$c] . "<br>";
        }
      }
      fclose($fp);
    ?>
  </BODY>
</HTML>
```

Funcția file_get_contents

Această funcție citește conținutul unui fișier de tip text și are doi parametri. Primul parametru este de tip șir de caractere și reprezintă numele fișierului al cărui conținut se va citi, iar cel de-al doilea parametru este opțional și indică dacă pentru găsirea fișierului se va folosi sau nu lista de directoare predefinită.

Funcția returnează un șir de caractere care conține toate datele din fișierul primit ca parametru în cazul în care citirea s-a efectuat cu succes și valoarea logică **`FALSE`** în caz contrar.

Funcția file

Această funcție citește conținutul unui fișier de tip text și are aceeași sintaxă cu funcția `file_get_contents`, dar rezultatul returnat de aceasta este un tablou de șiruri de caractere, fiecare dintre acestea constând într-o linie din fișierul primit ca parametru.

Funcția filetype

Această funcție returnează un șir de caractere care reprezintă tipul unui fișier. Funcția `filetype` are un singur parametru de tip șir de caractere, mai exact numele fișierului pentru care dorim să determinăm tipul. Valorile posibile pentru tipul unui fișier sunt `"pipe"`, `"fifo"`, `"char"`, `"dir"`, `"block"`, `"link"`, `"file"` și `"unknown"`.

Funcția filesize

Această funcție se folosește pentru a afla dimensiunea, exprimată în octeți, a unui fișier. Funcția `filesize` are un singur parametru de tip șir de caractere, mai exact numele fișierului pentru care dorim să determinăm dimensiunea.

Funcția fwrite

Această funcție se folosește pentru a scrie date într-un fișier text sau binar.

Pentru compatibilitate cu toate sistemele de operare, se recomandă ca la deschiderea fișierului, pentru scriere, să se adauge caracterul `"b"` la parametrul care reprezintă modul de deschidere a fișierului.

Funcția `fwrite` are trei parametri. Primul parametru este de tip întreg și reprezintă identificatorul de acces la fi-

șierul în care se dorește să se scrie date, cel de-al doilea parametru este de tip șir de caractere și reprezintă datele care se vor scrie în fișier, iar cel de-al treilea parametru este de tip întreg, opțional și reprezintă numărul de octeți din șirul de caractere care se vor scrie în fișier.

Funcția returnează un număr întreg care reprezintă numărul de octeți pe care a reușit să-i scrie în fișier, în cazul în care nu au apărut erori și valoarea logică **FALSE** în caz contrar.

În continuare prezentăm un exemplu de utilizare a acestei funcții și în figura alăturată poate fi observat efectul execuției codului:

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu fwrite
    </TITLE>
  </HEAD>
    <?PHP
      $fis = "c:\\test.txt";
      $sir = "Scriere in fisier";
      $fp = fopen($fis,"a");
      fwrite($fp,$sir);
      echo "Am scris ($sir) in fisierul ($fis).";
      fclose($fp);
    ?>
  </BODY>
</HTML>
```

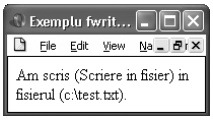

Figura 7: Funcția fwrite

Funcția fputs

Această funcție este o redenumire a funcției `fwrite`, pentru compatibilitate cu limbajul *C/C++*, și are același comportament. În limbajul *C/C++* funcția `fwrite` este folosită pentru a scrie un șir de octeți într-un fișier, spre deosebire de *PHP,* unde șirul de octeți este reprezentat chiar de șirul de caractere.

Funcția fflush

Această funcție forțează salvarea pe disc a conținutului zonelor tampon folosite pentru scrierea într-un fișier.

Funcția `fflush` primește un singur parametru de tip întreg, care reprezintă identificatorul de acces al fișierului pentru care se dorește golirea zonelor tampon.

Funcția returnează valoarea logică **FALSE** în caz de eroare și valoarea logică **TRUE** în caz contrar.

Funcția flock

Această funcție se folosește pentru a bloca sau debloca accesul la un fișier. Necesitatea utilizării funcției `flock` apare în momentul în care mai multe procese doresc să acceseze același fișier și nu se vrea afectarea integrității acestuia.

Funcția are trei parametri. Primul parametru reprezintă identificatorul de acces la conținutul fișierului, cel de-al doilea parametru reprezintă tipul operației de blocare a fișierului, iar al treilea parametru este de tip referință la întreg și va primi valorea logică **TRUE** în cazul în care funcția `flock` va opri execuția *script*-ului până în momentul în care va reuși să blocheze fișierul.

Pentru cel de-al doilea parametru, în limbajul *PHP* sunt disponibile constantele:

- `LOCK_SH` — se folosește în cazul în care fișierul se blochează în mod partajat pentru citirea datelor;
- `LOCK_EX` — se folosește în cazul în care fișierul se blochează în mod exclusiv pentru scrierea datelor;
- `LOCK_UN` — se folosește în cazul în care se dorește deblocarea unui fișier;
- `LOCK_NB` — se folosește împreună cu una dintre primele trei constante dacă nu se dorește ca funcția `flock` să întrerupă execuția *script*-ului până în momentul în care va reuși să efectueze operația cerută; această constantă se adună la una dintre primele trei.

Funcția returnează valoarea logică **TRUE** în caz de succes și valoarea logică **FALSE** în cazul unui eșec.

Funcția fseek

Această funcție se folosește pentru modificarea poziției de citire sau scriere în interiorul unui fișier.

Funcția `fseek` are trei parametri. Primul parametru este de tip întreg și reprezintă identificatorul de acces la un fișier, al doilea parametru este de tip întreg și indică numărul de poziții cu care se va face modificarea, iar al treilea parametru, tot de tip întreg, este opțional și specifică modul în care se va face modificarea.

Se observă că cel de-al doilea parametru poate avea și valori negative.

Modificarea poziției de citire sau scriere se poate face în trei moduri pentru care sunt definite constante:

- relativ la începutul fișierului — constanta `SEEK_SET`;
- relativ la poziția curentă — constanta `SEEK_CUR`;
- relativ la sfârșitul fișierului — constanta `SEEK_SET`.

În cazul în care cel de-al treilea parametru lipsește se va folosi ca valoare pentru acesta constanta `SEEK_SET`.

Funcția returnează o valoare întreagă egală cu `0`, în cazul în care nu au apărut erori, și egală cu -1 în caz contrar.

Această funcție nu poate fi folosită pentru fișiere care sunt accesate folosind unul dintre protocoalele *HTTP* sau *FTP*.

Funcția rewind

Această funcție schimbă poziția de citire sau scriere într-un fișier la începutul acestuia și are un singur parametru care specifică identificatorul de acces la fișier.

Rezultatul returnat de funcția `rewind` este același cu cel returnat de funcția `fseek`.

Apelul acestei funcții este echivalent cu apelul funcției `fseek` cu primul parametru având ca valoare identificatorul de acces al fișierului, al doilea parametru având valoarea `0` și cel de-al treilea parametru având valoarea `SEEK_SET` chiar dacă lipsește.

Funcția ftell

Această funcție se folosește pentru a afla poziția curentă de la care se va face următoarea citire sau scriere într-un fișier, relativ la începutul acestuia.

Funcția `ftell` primește ca parametru identificatorul de acces al fișierului pentru care se aplică și returnează o valoare întreagă reprezentând poziția din fișier, în caz de succes sau valoarea logică **`FALSE`** în caz de eșec.

Funcția ftruncate

Această funcție realizează redimensionarea fișierului cu o valoare primită ca parametru.

Funcția `ftruncate` are doi parametri. Primul dintre aceștia este identificatorul de acces al fișierului care se redimensionează, iar al doilea parametru este de tip întreg și reprezintă noua dimensiune a fișierului.

Trebuie ținut cont de faptul că în cazul în care cel de-al doilea parametru are o valoare mai mare decât dimensiunea curentă a fișierului va fi alocat, pe disc, spațiu suplimentar care nu va fi inițializat.

În cazul în care cel de-al doilea parametru are o valoare mai mică decât dimensiunea curentă a fișierului, atunci toate datele din fișier, care se află după poziția indicată de noua valoare, se vor pierde.

Funcția delete

Această funcție este mai degrabă o redenumire a funcției `unlink`, care se folosește pentru a șterge un fișier de pe disc.

Funcția `delete` a fost inclusă în pachetul de funcții oferit de limbajul *PHP* pentru cei care au programat în limbaje precum *Pascal* și nu au avut la dispoziție funcția `unlink`.

Funcția `unlink` primește ca parametru un nume de fișier și returnează valoarea logică **`TRUE`** în cazul în care ștergerea s-a efectuat cu succes.

În cazul în care ștergerea nu s-a putut efectua, funcția returnează valoarea logică **`FALSE`**.

Manipularea directoarelor

În cele ce urmează vom prezenta cele mai importante funcții și clase oferite de limbajul *PHP* pentru manipularea directoarelor.

Funcția getcwd

Această funcție returnează un șir de caractere care reprezintă directorul curent.

Funcția `getcwd` nu are nici un parametru.

În continuare prezentăm un exemplu de utilizare al acestei funcții al cărei rezultat poate fi observat în figura alăturată:

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu getcwd
    </TITLE>
  </HEAD>
  <BODY>
    <?PHP
      $cdir = getcwd();
      echo "Directorul curent este: ".$cdir;
    ?>
  </BODY>
</HTML>
```

Figura 8:
Funcția getcwd

Funcția chdir

Această funcție se folosește pentru a schimba directorul curent și primește ca parametru un șir de caractere care reprezintă noul director.

Funcția `chdir` returnează valoarea logică **`TRUE`** atunci când schimbarea directorului curent s-a efectuat cu succes și valoarea logică **`FALSE`** în caz contrar.

Funcția chroot

Această funcție se folosește pentru a schimba directorul rădăcină pentru procesul curent și primește ca parametru un șir de caractere care reprezintă noul director rădăcină.

Funcția `chroot` returnează valoarea logică **`TRUE`** atunci când schimbarea directorului rădăcină s-a efectuat cu succes și valoarea logică **`FALSE`** în caz contrar.

Această funcție nu este implementată pe platformele de operare *Microsoft Windows* și nu este recomandată folosirea ei decât în cazul în care interpretorul *PHP* este utilizat folosind metoda de execuție *CGI*.

Funcția opendir

Această funcție se folosește pentru a deschide un director în scopul citirii conținutului său.

Funcția `opendir` are un singur parametru de tip șir de caractere care reprezintă numele directorului al cărui conținut se va prelucra.

Această funcție returnează un identificator de acces la conținutul directorului și este de tipul resursă.

În cazul în care nu se reușește deshiderea directorului, funcția returnează valoarea logică **`FALSE`**.

Funcția closedir

Această funcție se folosește pentru a închide un director care a fost deschis pentru citire folosind funcția `opendir`.

Funcția `closedir` are un singur parametru de tipul resursă, care reprezintă identificatorul de acces al directorului care se dorește a fi închis.

Această funcție nu returnează nici o valoarea, ea având tipul **`void`**.

Funcția readdir

Funcția `readdir` se folosește pentru a citi următoarea intrare disponibilă într-un director.

Ea are un singur parametru de tip resursă, care reprezintă identificatorul de acces al directorului din care se va citi.

Funcția `readdir` returnează un șir de caractere reprezentând numele următoarei intrări din director, în cazul în care nu am ajuns la sfârșitul acestuia sau nu au apărut erori

și valoarea logică **FALSE** în caz contrar.

În continuare vă prezentăm un exemplu de utilizare a funcțiilor `opendir`, `readdir` și `closedir`, iar în figura alăturată poate fi observat efectul execuției codului următor:

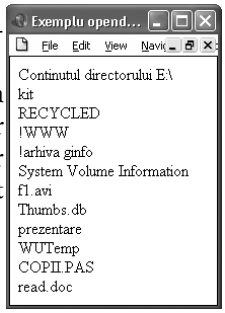

Figura 9: Funcțiile opendir, readdir, closedir

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu opendir,
            readdir,closedir
    </TITLE>
  </HEAD>
    <?PHP
      $dir = "E:\\";
      $dh = opendir($dir);
      echo "Continutul directorului $dir<BR>";
      while (false !== ($fis = readdir($dh))){
        echo "$fis<BR>";
      }
      closedir($dh);
    ?>
  </BODY>
</HTML>
```

Funcția rewinddir

Această funcție resetează poziția curentă dintr-un director, și anume, dacă după un anumit număr de citiri este apelată această funcție și după aceeea se mai citește o intrare din directorul respectiv, atunci se va returna prima intrare din director.

Funcția are un singur parametru de tipul resursă, care reprezintă identificatorul de acces al directorului din care se va citi și nu returnează nici o valoare.

Clasa dir

Această clasă încapsulează funcțiile `opendir`, `closedir`, `readdir` și `rewinddir`. Structura acesteia este următoarea:

```
class dir{
  dir(string director);
  string path;
  resource handle;

  string read();
  void rewind();
  void close();
}
```

Constructorul clasei `dir` primește ca parametru un șir de caractere care reprezintă numele directorului care se va citi și apelează funcția `opendir`.

Funcțiile `read`, `rewind` și `close` din cadrul acestei clase au același comportament cu funcțiile `readdir`, `rewinddir`, respectiv `closedir` și se deosebesc de acestea prin simplul fapt că nu mai au parametri, deoarece identificatorul de acces la director este de asemenea încapsulat în cadrul clasei `dir`.

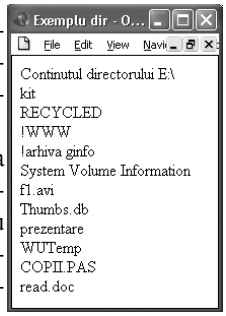

Figura 10: Clasa dir

În continuare vă vom prezenta un exemplu de utilizare a acestei clase care reprezintă implementarea cu clase a exemplului anterior și rezultatul execuției codului poate fi observat în figura 10.

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu dir
    </TITLE>
  </HEAD>
    <?PHP
      $d = dir("E:\\");
      echo "Continutul directorului $d->path
                                          <BR>";
      while (false !== ($fis = $d->read())){
        echo "$fis<BR>";
      }
      $d->close();
    ?>
  </BODY>
</HTML>
```

Funcția mkdir

Această funcție se folosește pentru a crea un director nou și primește ca parametru un șir de caractere care reprezintă numele directorului care se va crea și atributele acestuia (parametru de tip întreg) în format **nix*.

Funcția returnează valoarea logică **TRUE** în caz de succes și valoarea logică **FALSE** în caz de eșec.

Funcția rmdir

Această funcție se folosește pentru a șterge un director de pe disc și primește ca parametru un șir de caractere care reprezintă numele directorului care va fi șters.

Directorul care se dorește a fi șters trebuie să nu conțină nici un fișier sau subdirector.

Funcția returnează valoarea logică **TRUE** în caz de succes și valoarea logică **FALSE** în caz de eșec.

Alte funcții

În cele ce urmează vom prezenta un set de funcții care facilitează lucrul cu fișiere și/sau directoare .

Funcția basename

Această funcție returnează numele de bază al unui fișier sau director sub forma unui șir de caractere.

Funcția are doi parametri de tip șir de caractere. Primul parametru reprezintă o cale către un fișier sau director, iar al doilea este opțional și reprezintă sufixul fișierului.

În continuare vă prezentăm un exemplu de utilizare a acestei funcții și în figura alăturată se poate observa efectul execuției codului următor:

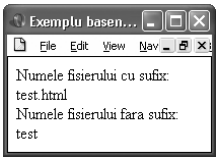

Figura 11:
Funcția basename

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu basename
    </TITLE>
  </HEAD>
    <?PHP
      $cale = "c:\\test.html";
      echo "Numele fisierului cu sufix:<BR>";
      echo basename($cale)."<BR>";
      echo "Numele fisierului fara sufix:<BR>";
      echo basename($cale,".html");
    ?>
  </BODY>
</HTML>
```

Funcția dirname

Această funcție returnează numele directorului părinte al unui fișier sau director sub forma unui șir de caractere.

Funcția are un singur parametru de tip șir de caractere. care reprezintă o cale către un fișier sau director.

În continuare vă prezentăm un exemplu de utilizare a acestei funcții și în figura alăturată se poate observa efectul execuției codului următor:

Figura 12:
Funcția dirname

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu dirname
    </TITLE>
  </HEAD>
    <?PHP
      $cale = "c:\\test.html";
      echo "Directorul parinte pentru  ";
      echo  $cale." este ".dirname($cale);
    ?>
  </BODY>
</HTML>
```

Funcția pathinfo

Această funcție se folosește pentru a afla informații referitoare la o cale către un fișier sau director.

Funcția `pathinfo` primește ca parametru un șir de caractere care reprezintă calea către un fișier sau director.

Funcția returnează un tablou ale cărui valori sunt de tipul șir de caractere și care conține elementele `"dirname"`, `"basename"` și `"extension"`.

Elementele `"dirname"` și `"basename"` au valori identice cu cele returnate de funțiile cu același nume, iar elementul `"extension"` conține extensia fișierului.

În continuare vă prezentăm un exemplu de utilizare a acestei funcții și în figura 13 se poate observa efectul execuției codului următor:

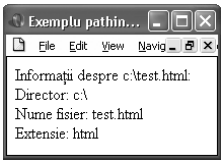

Figura 13:
Funcția pathinfo

```
<HTML>
  <HEAD>
    <TITLE>
      Exemplu pathinfo
    </TITLE>
  </HEAD>
    <?PHP
      $cale = "c:\\test.html";
      $info = pathinfo($cale);
      echo "Informații despre $cale:<BR>";
      echo "Director: ";
      echo $info["dirname"]."<BR>";
      echo "Nume fisier: ";
      echo $info["basename"]."<BR>";
      echo "Extensie: ";
      echo $info["extension"]."<BR>";
    ?>
  </BODY>
</HTML>
```

Funcția file_exists

Această funcție se folosește pentru a determina dacă un fișier există sau nu pe disc.

Funcția `file_exists` are un singur parametru de tip șir de caractere care reprezintă numele fișierului pentru care se verifică existența fizică pe disc.

Funcția returnează valoarea logică **`TRUE`** în caz de succes și valoarea logică **`FALSE`** în caz de eșec.

Această funcție nu poate fi folosită pentru fișiere aflate la distanță decât dacă sunt partajate de pe o platformă *Microsoft Windows* pe care a fost configurată componenta *File and Print Sharing* sau dacă sunt partajate pe o platformă **nix* pe care a fost instalat pachetul *Samba*.

Funcția copy

Această funcție se folosește pentru a copia un fișier și are doi parametri de tip șir de caractere. Primul dintre aceștia reprezintă numele fișierului sursă și cel de-al doilea reprezintă numele fișierului destinație.

Funcția returnează valoarea logică **`TRUE`** în caz de succes și valoarea logică **`FALSE`** în caz de eșec.

Funcția rename

Această funcție se folosește pentru a redenumi un fișier și are doi parametri de tip șir de caractere. Primul parametru reprezintă numele fișierului și cel de-al doilea reprezintă noul numele al acestuia.

Funcția returnează valoarea logică **`TRUE`** în caz de succes și valoarea logică **`FALSE`** în caz de eșec.

Va urma...

În articolul următor vom prezenta în detaliu câteva facilități importante oferite de limbajul *PHP*, dintre care amintim interoperabilitatea acestui limbaj cu platforma *JAVA* și crearea dinamică a imaginilor pentru publicarea pe diferite *site*-uri.