Lekcja 10

Uprawnienia

Aby skrypt PHP mógł odwołać się do pliku, musi mieć odpowiednie uprawnienia. Szczegóły są zależne od serwera. Najczęściej chyba skrypt ma uprawnienia takie, jak użytkownik który go umieścił. Może też mieć uprawnienia użytkownika nobody - wtedy żeby móc ze skryptu zmodyfikować jakiś plik, trzeba nadać prawo zapisu wszystkim użytkownikom. Dokładne informacje na ten temat powinien dostarczyć administrator serwera na którym umieszczamy skrypty (prawa dostępu do pliku jak w linuksie – polecenie chmod).

Dołączanie plików przy pomocy funkcji include()

Funkcja include() pozwala na dołączanie plików do dokumentów PHP. Kod wykonywany będzie w taki sam sposób jak gdyby znajdował się w głównym dokumencie.

Przykład:

```
<?php
include(„message.php”);
?>
```

plik message:

```
<?php
Echo „tekst dołączony”;
?>
```

Można także użyć funkcji include() w pętli. Wówczas przy każdym obiegu pętli będzie wykonywana zawartość pliku dołączonego.

Sprawdzanie, czy plik istnieje przy pmocy funkcji file_exists()

Argument tej funkcji wymaga podania względnej lub bezwzględnej ścieżki do testowanego pliku. Jeśli plik istnieje to funkcja zwróci wartość true, w przeciwnym razie będzie to wartość false. Przykład:

```
If (file_exists("test.txt”))
Print “Plik istnieje”;
```

Przy pomocy funkcji is_file(); można sprawdzić czy testowany “obiekt” jest plikiem. Argumentem jest ścieżka dostępu do pliku. Podobnie sprawdzamy czy obiekt testowany jest katalogiem. Służy do tego funkcja is_dir(), która również wymaga podania ścieżki. Przykład:

```
if (is_dir("/kat1"))
print "/kat1 jest katalogiem";
```

Funkcje sprawdzające status pliku:

is_readable() – funkcja informująca, czy użytkownik może czytać plik;
is_writeable() – funkcja informująca, czy użytkownik może pisać do pliku;
is_executable() - funkcja informująca, czy użytkownik może wykonywać plik;
filesize() – określa wielkość pliku w bajtach;
fileatime() – data ostatniego dostępu do pliku;
filemtime() – data ostatniej modyfikacji pliku;
filectime() – data ostatniej modyfikacji dokumentu (systemy UNIX, na innych platformach data utworzenia),
filegroup(nazwa) - zwraca identyfikator grupy, do której należy plik;
fileowner(nazwa) - zwraca identyfikator właściciela pliku;
ftell($file) - zwraca pozycję w otwartym pliku;
mkdir(nazwa) - tworzy katalog o podanej nazwie;
readfile(nazwa) - wyświetla zawartość pliku;

Przykład użycia funkcji filemtime():

```
<?php
$mdata=filemtime("test.txt");
Print “ostatnia modyfikacja pliku test.txt była”;
Print date("D d M Y g:I A",$mdata);
```

Tworzenie i usuwanie plików

Jeśli plik jeszcze nie istnieje to możemy go utworzyć za pomocą funkcji **touch()**. Jeśli okaże się, że plik, który chcemy utworzyć już istniej, wówczas zostanie zmieniona data modyfikacji tego pliku bez ingerencji w jego zawartość. W celu usunięcia pliku wykonujemy funkcję **unlink()**. Przykład:

```
unlink("mojplik.txt");
```

Otwieranie plików

Przed wykonaniem jakiejkolwiek operacji na pliku, należy go otworzyć. Służy do tego funkcja fopen(). Pobiera ona 2 argumenty - pierwszy to nazwa pliku który chcemy otworzyć. Drugi parametr może mieć następujące wartości:
r - otwiera plik do odczytu
r+ - otwiera plik do odczytu i zapisu
w - kasuje zawartość pliku i otwiera go do zapisu
w+ - kasuje zawartość pliku i otwiera go do zapisu i odczytu
a - otwiera plik do dopisywania
a+ - otwiera plik do dopisywania i odczytu
Co ciekawe, funkcja fopen() może otworzyć plik na innym serwerze przez protokół http (tyl-

ko odczyt), lub ftp (odczyt lub zapis, ale nie jednocześnie). fopen() zwraca liczbę, która służy do identyfikowania otwartego pliku. Przykład:

```
$file=fopen("file.txt", "r");      // otwiera plik file.txt do odczytu
$file=fopen("file.txt", "w");      /* otwiera plik do zapisu. Jeżeli plik nie istnieje,
                                      zostanie utworzony */
$file=fopen("ftp://adres.pl/plik", "r");   // otwiera plik przez protokół ftp
```

Po zakończeniu operacji na pliku, należy go zamknąć funkcją fclose, np:

```
$file=fopen("file.txt", "r");
fclose($file);
```

Odczyt z pliku

Jest kilka funkcji służących do odczytu z pliku. Zacznę może od funkcji fgetc - odczytuje ona 1 znak z pliku:

```
$znak=getc($file);
```

Jeśli funkcja napotka koniec pliku, zwróci wartość FALSE. Po odczytaniu znaku, pozycja w pliku jest przesuwana o 1 do przodu. Dzięki temu, po kolejnym wywołaniu funkcji getc otrzymamy kolejny znak, a nie jeszcze raz ten sam. Kolejną funkcją jest funkcja fgets:

```
$linia=fgets($file, $maxLen);
```

Drugi parametr określa ile znaków funkcja może odczytać. Trzecią funkcją do odczytu z pliku jest fread. Działa ona podobnie do fgets, z tym że nie przerywa czytania gdy natrafi na znak nowej lini.

Pomimo, iż już jesteśmy w stanie czytać wiersze przy pomocy funkcji fgets(), potrzebujemy czasami sposobu na stwierdzenie, czy nie został napotkany koniec pliku. Służy do tego funkcja feof(), zwracająca wartość true w momencie osiągnięcia końca pliku oraz false w przeciwnym przypadku:

```
$koniec=feof($file);
```

Przykład: otwarcie i odczyt pliku wiersz po wierszu

```
<?php
$nazwapliku="test.txt";
$file = fopen($nazwapliku, "r") or die("Nie mogę otworzyć pliku nazwapliku");
While (!feof($file))
{
$wiersz = fgets($file,1024);
Print "$wiersz <BR>";
}
?>
```

Zamiast czytać plik wiersz po wierszu, można odczytywać go we fragmentach o zadanej wielkości. Funkcja fread() wymaga podania w argumencie uchwytu pliku oraz ilości bajtów jaka ma być odczytana. Funkcja zwraca ilość wymaganych danych, o ile wcześniej nie zostanie napotkany znak końca pliku. Jeśli chcemy określić pozycję, od której ma rozpocząć się pobieranie danych, wówczas wykorzystujemy funkcję fseek(). Wymaga ona dwóch argumentów: uchwytu pliku oraz liczby całkowitej reprezentującej przesunięcie wskaźnika (w bajtach). Przykład poniżej demonstruje użycie funkcji fseek() oraz fread() do wyświetlenia w oknie przeglądarki zawartości drugiej części pliku:

```
<?php
$nazwapliku="test.txt";
$file = fopen($nazwapliku, "r") or die("Nie mogę otworzyć pliku $nazwapliku");
$rozmiar = filesize($nazwapliku);
$polowa = (int)($rozmiar / 2);
print "Połowa pliku zaczyna się od bajtu: $polowa );
$kawalek = fread fread($file, ($rozmiar-$polowa));
print $kawalek;
?>
```

Zapis do pliku

Do zapisywania w pliku służy funkcja fwrite:

```
fwrite($file, $tekst, $maxLen);
```

Funkcja ta zapisuje tekst podany jako 2 parametr do pliku identyfikowanego przez $file. 3 parametr jest opcjonalny i określa maksymalną długość tekstu, jaka może zostać zapisana. Jest jeszcze funkcja fputs, ale działa ona dokładnie tak samo. W PHP często występują identyczne funkcje pod innymi nazwami - ułatwia to życie osobom przyzwyczajonym do innych języków.

Blokowanie plików

Jak dotąd wszystko wydaje się proste. Co jednak stanie się, jeżeli jednocześnie zostaną uruchomione dwie kopie skryptu, i obie będą próbowały zmienić ten sam plik? Sytuacja taka jak łatwo się domyślić, może mieć przykre konsekwencje. Aby rozwiązać ten problem należy użyć blokowania plików. Służy do tego funkcja flock:

```
flock($file, mode);
```

Pierwszy argument funkcji flock to identyfikator otwartego pliku. Argument mode określa typ dostępu, jaki chcemy uzyskać. Może mieć następujące wartości:
1 - dostęp do odczytu
2 - dostęp do zapisu
3 - zwolnienie blokady

Jeżeli chcemy uzyskać dostęp do zapisu, najpierw wszystkie inne blokady muszą być zwolnione. Do uzyskania dostępu do odczytu wystarczy, ze plik nie będzie zablokowany do zapisu - w ten sposób wiele skryptów może jednocześnie czytać z pliku, ale tylko jeden może do niego zapisywać. Dodatkowo w trakcie zapisu żaden inny skrypt nie może odczytywać z pliku - dzięki temu nie natrafi na niekompletne dane.

Jeżeli określony rodzaj dostępu nie może być w danej chwili uzyskany, wykonanie skryptu zostanie wstrzymane do czasu, gdy będzie to możliwe. Plik powinien być blokowany na jak najkrótszy czas, aby nie wstrzymywać niepotrzebnie innych operacji. Jeżeli chcemy tylko sprawdzić, czy plik jest zablokowany, do argumentu mode należy dodać 4 - wtedy funkcja zwróci wartość TRUE jeżeli plik nie jest zablokowany, lub FALSE gdy jest. Przykład:

```
$file1=fopen("jakis.plik", "r");      // otwiera 2 razy ten sam plik
$file2=fopen("jakis.plik", "r");

flock($file1, 2);                  // blokuje pierwszą kopie

if(flock($file2, 6))   /* funkcja flock zwróci false, ponieważ plik jest już zablokowany */
{
  echo("Plik nie zablokowany");
}else{
  echo("Plik zablokowany");
}
flock($file, 3);            /* odblokowuje plik */
```

Inne funkcje

PHP zawiera wiele funkcji służących do obsługi plików i katalogów. Oto niektóre z nich:
mkdir() – tworzy katalog;
rmdir() – usuwakatalog;
opendir() – otwarcie katalogu (funkcja niezbędna do przeglądania katalogu)

readdir() – odczyt nazwy pliku lub podkatalogu

Przykład demonstrujący przedstawienie zawartości katalogu:

```
<?php
$nazwakatalogu = "kat1";
$kh = opendir($nazwakatalogu);
while (gettype($plik = readdir ($kh))!=Boolean)
{
if (is_dir("$nazwakatalogu/$plik"))
print "(D) "
print "$plik<BR>"
}
closedir($kh);
?>
```

Prosty licznik tekstowy

Licznik tekstowy posłuży jako praktyczny przykład wykorzystania dostępu do plików:

```
if(file_exists("counter.n"))              // sprawdza, czy plik istnieje
{
  $file=fopen("counter.n", "r");      // otwiera plik
  flock($file, 1);                          // blokuje plik

  $ile=fgets($file, 100);                // odczytuje wartość

  flock($file, 3);               // odblokowuje plik
  fclose($file);              // zamyka plik

  $ile++;                    // zwiększa wartość o 1
}
else
  $ile=1;                                /* jeśli plik nie istnieje,  wyświetli się 1 */

$file=fopen("counter.n", "w");       // otwiera plik do zapisu
flock($file, 2);                          // blokuje do zapisu

fwrite($file, $ile);              // zapisuje wartość

flock($file, 3);             // odblokowuje plik
fclose($file);                  // zamyka plik

echo($ile);                // wyświetla wartość
```

Zadania

Zad.1

Napisz skrypt, który korzysta z pliku dołączonego i wyświetla dowolny komunikat.

Zad. 2

Napisz skrypt, który wykorzysta wszystkie poznane w tej lekcji funkcje służące do odczytywania informacji o plikach (status pliku).

Zad. 3

Utwórz plik tekstowy w katalogu, w którym zapisywane są pliki *.php. Napisz skrypt, który odczyta zawartość tego pliku.

Zad. 4

Napisz skrypt, który będzie dopisywał do pliku utworzonego w zadaniu 3 tekst wpisany w oknie formularza.

Zad. 5

Napisz skrypt, który odczyta fragment pliku (np. fragment od 25 do 50 bajtu).

Zad. 6

Napisz skrypt przedstawiający zawartość katalogu (wcześniej należy utworzyć katalog oraz utworzyć w nim podkatalogi oraz pliki).

Zad. 7

Napisz skrypt pokazujący liczbę odwiedzi na Twojej stronie.

Zad. 8

Utwórz formularz do którego użytkownik wprowadzi imię i nazwisko. Następnie utwórz skrypt zapisujący te dane do pliku.

Zad. 9

Napisz skrypt czytający dane, które zapisano w zadaniu 8 oraz wypisujący je w oknie przeglądarki. Umieśc także informacje o ilości wierszy w pliku oraz jego rozmiarze.

Pytania

1. Jakiej funkcji należy użyć, by dołączyć do uruchamianego skryptu kod biblioteki?
2. Jakiej funkcji należy użyć, chcąc stwierdzić czy dany plik istnieje?
3. Jak określić wielkość pliku?
4. Jakiej funkcji należy użyć do otwarcia pliku do czytania i pisania?
5. Jakiej funkcji należy użyć by odczytać z pliku wiersz danych