Funkcje daty i czasu

Język PHP dysponuje dużą grupą funkcji, których zadaniem jest wykonanie operacji na dacie i czasie.

Funkcja time() zwraca informacje na temat bieżącej daty i czasu. Funkcja nie posiada żadnych parametrów. Informacje na temat daty i czasu są zwracane w postaci liczby. Liczba ta odpowiada liczbie sekund, które upłynęły od godziny 00:00:00 1.01.1970 roku do bieżącej daty. Jest to tak zwany znacznik_czasu(timestamp)

```php
<?php
echo time();

?>
```

```
1428769635
```

Taki zapis daty i czasu pozwala na wykonanie działań na dacie, np.. zwiększanie daty o jeden dzień, czasu o jedną godzinę itd..

Funkcja getdate()

**Funkcja getdate(opcjonalnie znacznik_czasu) -wynikiem działania funkcji jest tablica asocjacyjna zawierająca dane dotyczące daty i czasu.
Indeksy oraz wartości tej tablicy dotyczą poszczególnych elementów wchodzących w skład daty i czasu. Argument znacznik_czasu jest opcjonalny. Jeżeli nie zostanie podany, działania wykonane przez funkcję bedą dotyczyły bieżącej daty.**

```php
<?php
var_dump(getdate());
?>
```

```
array(11) { ["seconds"]=> int(36) ["minutes"]=> int(37) ["hours"]=> int(18) ["mday"]=> int(11) ["wday"]=> int(6) ["mon"]=> int(4) ["year"]=> int(2015) ["yday"]=> int(100) ["weekday"]=> string(8) "Saturday" ["month"]=> string(5) "April" [0]=> int(1428770256) }
```

Funkcja getdate()

Klucze zwróconej tablicy asocjacyjnej

Klucz	Opis	Przykłady zwracanych wartości
"seconds"	Ilość sekund	*0* do *59*
"minutes"	Liczba minut	*0* do *59*
"hours"	Ilość godzin	*0* do *23*
"mday"	Liczba będąca dniem miesiąca	*1* do *31*
"wday"	Dzień tygodnia w postacy cyfry	*0* (dla Niedzieli) aż do *6* (dla Soboty)
"mon"	Miesiąc w postaci liczby	*1* aż do *12*
"year"	Pełen rok w postaci liczby, 4 cyfry	Przykłady: *1999* lub *2003*
"yday"	Dzień danego roku, w postaci liczby	*0* aż do *365*
"weekday"	Pełna nazwa dnia tygodnia, w języku angielskim	*Sunday* aż do *Saturday*
"month"	Pełna nazwa miesiąca, w języku angielskim, jak np. January lub March	*January* aż do *December*
0	Sekundy, które upłynęły od Ery Uniksa, podobnie jak wartość zwracana przez **time()** i użyta przez funkcję **date()**.	

Wykorzystanie funkcji getdate() do wyświetlenia bieżącej daty

```
1  <!DOCTYPE html>
2  <html lang="pl">
3  <head>
4  <title>bieżąca data</title>
5  <meta charset="utf-8">
6  </head>
7  <body>
8
9  <?php
10 $data=getdate();//$data jest tablica asocjacyjna
11 $dzien=$data['mday'];
12 if ($dzien<10) {$dzien='0'.$dzien;}
13 $miesiac=$data['mon'];
14 if ($miesiac<10) {$miesiac='0'.$miesiac;}
15 $rok=$data['year'];
16 echo 'bieżąca data to: '.$dzien.'-'.$miesiac.'-'.$rok.'r';
17 ?>
18 </body>
19 </html>
```

bieżąca data to: 11-04-2015r

Funkcja date()

Funkcją, która pozwala na formatowanie w odpowiedni sposób daty i czasu, jest funkcja

date(format [,znacznik_czasu])

Parametr format określa, jakie informacje na temat daty i czasu powinny zostać zwrócone. Parametr format jest ciągiem znaków, składającym się ze znaczników.

```
1   <!DOCTYPE html>
2   <html lang="pl">
3   <head>
4   <title>funkcja date()</title>
5   <meta charset="utf-8">
6   </head>
7   <body>
8
9   <?php
10  echo date('d-m-Y');
11  ?>
12  </body>
13  </html>
```

11-04-2015

Funkcja date() - niektóre znaczniki formatujace

d - Dzień miesiąca, 2 cyfry z wiodącymi zerami

D - Tekstowy opis angielskiej nazwy dnia, trzy litery

j - Dzień miesiąca bez zer wiodących

l (mała litera 'L') - Pełen angielski opis dnia tygodnia

N - Liczbowa forma dnia tygodnia, zgodna z normą ISO-8601 (dodana w PHP 5.1.0)

S - Angielski przyrostek porządkowy dla dnia miesiąca, 2 litery

w - Liczbowa forma dnia tygodnia

z - Dzień roku (Zaczynając od 0)

W - Numer tygodnia w roku, zgodny z normą ISO-8601, Tygodnie rozpoczynają Poniedziałki (dostępne od PHP 4.1.0)

F - Pełen angielski opis, dnia miesiąca, taki jak January czy March

m - Liczbowa forma miesiąca, z zerami wiodącymi

M - Krótki, angielski opis miesiąca, trzy litery

Funkcja date() - niektóre znaczniki formatujące

n - Liczbowa forma miesiąca, bez zer wiodących

t - Ilość dni w danym miesiącu

L - Informacja o tym, czy rok jest przestępnym

o - Numer roku, zgodny z normą ISO-8601. Zwraca to taką samą wartość jak Y, z takim wyjątkiem, że numer tygodnia ISO (W) należy do poprzedniego lub następnego roku, niż rok użyty w tym miejscu. (dodane w PHP 5.1.0)

Y - Pełna liczbowa forma roku, 4 cyfry

y - Dwie cyfry reprezentujące rok

a - Pora dnia - dwie małe litery (przed/po południu) (ang. Ante/Post meridiem)

A - Pora dnia - dwie duże litery (przed/po południu) (ang. Ante/Post meridiem)

g - Godzina, w formacie 12-godzinnym, bez zer wiodących

G - Godzina, w formacie 24-godzinnym, bez zer wiodących

h - Godzina, w formacie 12-godzinnym, z zerami wiodącymi

H - Godzina, w formacie 24-godzinnym, z zerami wiodącymi

Funkcja date() - niektóre znaczniki formatujace

i - Minuty z zerami wiodącymi

s - Sekundy, z zerami wiodącymi

e - Identyfikator strefy czasowej (dodano w PHP 5.1.0)

I (duże i) - Informacja o tym, czy czas jest letni

O - Różnica z czasem Greenwich (GMT) w godzinach

P - Różnica z czasem Greenwich (GMT) z dwukropkiem pomiędzy godzinami i minutami (dodano w PHP 5.1.3)

T - Skrót dla strefy czasowej

Z - Różnica dla strefy czasowej w sekundach. Wyrównanie to jest zawsze ujemne dla stref położonych na zachód od południka 0, oraz dodatnie dla tych leżących na wschód od niego.

c - Data w standardzie ISO 8601 (dodana w PHP 5)

r - Data sformatowana zgodnie z RFC 2822

U - Sekundy liczone od ery UNIX-a (1 stycznia 1970 00:00:00 czasu Greenwich - GMT)

Funkcja mktime()

Funkcja mktime() zwraca znacznik czasu daty podanej jako argument funkcji. Funkcja może posiadać od 0 do 6, argumentów podanych w postaci liczb całkowitych.
Są to kolejno:
Godzina, minuta, sekunda, miesiąc, dzien miesiąca, rok

Znacznik czasu, który zwróci funkcja mktime() może być wykorzystany w funkcji getdate() i date()

```php
<?php
$znacznikCzasu=mktime(13,23,30,12,01,2013);
echo $znacznikCzasu.'<br>';
var_dump(getdate($znacznikCzasu));
?>
```

```
1385900610
array(11) { ["seconds"]=> int(30) ["minutes"]=> int(23) ["hours"]=> int(13)
["mday"]=> int(1) ["wday"]=> int(0) ["mon"]=> int(12) ["year"]=> int(2013)
["yday"]=> int(334) ["weekday"]=> string(6) "Sunday" ["month"]=> string(8)
"December" [0]=> int(1385900610) }
```

Funkcje formatowania ciągów
Funkcja nl2br()

Jeżeli wyświetlamy w przeglądarce blok tekstu, który zawiera znak końca linii, to przeglądarka nie uwzględni tych znaków.

**Funkcja nl2br('ciąg znaków') dla wybranego bloku tekstu, przed każdym znakiem końca linii automatycznie wstawi znacznik
**

Zastosowanie funkcji nl2br()

W mądrość Jasia wierząc skrycie, Pytał chłopca nauczyciel O powstańców polskich liczne Wpływy ideologiczne. Cisza. Rozpacz. Pustka w głowie. Więcej zgoła nic, albowiem Pozbawionym wszelkiej wiedzy Nie pomogą i koledzy. Czasem tylko dusza załka - Lecz nie chłopca, a Marszałka. Marszczy belfer brwi okropnie. Gdybyż pytał nie na stopnie(!)

użycie funkcji nl2br()

W mądrość Jasia wierząc skrycie,
Pytał chłopca nauczyciel
O powstańców polskich liczne
Wpływy ideologiczne.

Cisza. Rozpacz. Pustka w głowie.
Więcej zgoła nic, albowiem
Pozbawionym wszelkiej wiedzy
Nie pomogą i koledzy.

Czasem tylko dusza załka -
Lecz nie chłopca, a Marszałka.
Marszczy belfer brwi okropnie.
Gdybyż pytał nie na stopnie(!)

```
7  <body>
8  <?php
9  $ciagznakow='W mądrość Jasia wierząc skrycie,
10 Pytał chłopca nauczyciel
11 O powstańców polskich liczne
12 Wpływy ideologiczne.
13
14     Cisza. Rozpacz. Pustka w głowie.
15     Więcej zgoła nic, albowiem
16     Pozbawionym wszelkiej wiedzy
17     Nie pomogą i koledzy.
18
19 Czasem tylko dusza załka -
20 Lecz nie chłopca, a Marszałka.
21 Marszczy belfer brwi okropnie.
22 Gdybyż pytał nie na stopnie(!)';
23
24 echo $ciagznakow.'<br>';
25 echo '<h2>użycie funkcji nl2br() </h2>';
26 echo nl2br($ciagznakow);
27 ?>
28 </body>
```

Sekwencje sterujące

Wywołanie instrukcji:

```
echo "Statek"Titanic"rozbil się o gore lodowcową";
```

Spowoduje błąd, ponieważ wewnątrz cudzysłowów nie możemy wstawiać cudzysłowów.

Rozwiązaniem tego problemu są sekwencje sterujące, są one połączeniem ukośnika (\) i znaku który ma zostać wstawiony:

- \" kolejny znak zostanie wstawiony jako cudzysłów, nie traktowany jako koniec znacznika
- \' kolejny znak zostanie wstawiony jako spostrof
- \\ kolejny znak zostanie wstawiony jako ukośnik
- \$ kolejny znak zostanie wstawiony jako $, nie traktowany jako część nazwy zmiennej

Poprawne wywołanie :

```
echo "Statek \"Titanic\" rozbil się o gore lodowcowa,
    kosz biletu 20\$ "
```

Składnia heredoc

Łańcuch znaków można utworzyć na trzy sposoby.

- pojedyncze cudzysłowy
- podwójne cudzysłowy
- składnia heredoc

```
<<<ogranicznik

    Tresc

ogranicznik;
```

Jeszcze jednym sposobem na zapisanie łańcucha znaków jest użycie skladni heredoc ("<<<"). Po operatorze <<< powinno się umieścić identyfikator i takim samym identyfikatorem trzeba zakończyć łańcuch znaków.

Identyfikator zamykający *musi* zaczynać się w pierwszej kolumnie nowej linii. Identyfikator musi też podlegać regułom nazewnictwa w PHP: musi się składać wyłącznie z alfanumerycznych znaków oraz znaku podkreślenia i musi zaczynać się od litery lub znaku podkreślenia.

Ostrzeżenie

Ważne by pamiętać, że linia zawierająca identyfikator zamykający nie może zawierać żadnych innych znaków, z *wyjątkiem* średnika (;). Znaczy to przede wszystkim, że identyfikator zamykający *nie może być wcinany*, i nie może być żadnych spacji ani tabulacji przed lub za średnikiem.

Najbardziej dokuczliwym ograniczeniem jest to, że wewnątrz tego łańcucha znaków nie może być znaku powrotu karetki (\r) na końcu linii, jedynie znak nowej linii (\n). Ponieważ Microsoft Windows używa jako znaku końca linii sekwencji \r\n, łańcuchy znaków zapisne w składni heredoc mogą nie działać, jeśli skrypt zostanie napisany w edytorze windowsowym. Na szczęście większość edytorów tekstowych udostępnia możliwość zapisania pliku w uniksowym formacie końca linii.

Składnia heredoc zachowuje się podobnie jak tekst w cudzysłowach podwójnych. W tej składni nazwa zmiennej zamieniana jest na jej wartość, ale należy zachować ostrożność przy zapisie złożonych zmiennych razem z tekstem.

Heredoc

```
9   $str=<<<ABC
10  tutaj dowolny ciąg znaków
11
12  ABC;
```

```
8   <?php
9   $zmienna = "tresc";
10  $str=<<<jakisOgranicznik
11  mozemy wewnatrz uzywac "cudzysłowów" <br>
12      Jak i 'apostrofów'
13      c:\\
14      Zmienna \$zmienna zawiera tresc $zmienna
15  Możemy napisac jakisOgranicznik ale nic się nie stanie
16  Należy napisac jakisOgranicznik w nowym wierszu by skonczyc :
17
18  jakisOgranicznik;
19  //kolejny tekst musi się zaczynać od nowej linii
20  echo $str;
21  ?>
```

- Uzyjemy <<< by poinformować PHP że będziemy korzystać z trybu heredoc.
- Możemy użyć dowolnego ogranicznika
- Możemy używać cudzysłowów i apostrofów
- Zmienne są interpretowane, więc musimy używać sekwencji sterujących by wypisać znak $

Funkcje formatowania ciągów
Funkcja wordwrap()

string wordwrap (string $str [, int $szerokość [, string $break [, bool $cut]]])

Do formatowania tekstu w postaci kolumny o określonej szerokości można wykorzystać funkcję wordwrap(). Dzieli ona ciąg podany jako argument na linie o maksymalnej długości 75 znaków. Do rozdzielania linii domyślnie jest używany znak „\n”.

Jeżeli $cut jest ustawiony na TRUE, łańcuch jest zawsze łamany w określonej szerkości. Gdy mamy wyraz, który jest dłuższy od podanej szerokości, zostanie on przełamany.

$break-ciąg znaków, który zostanie zastosowany do dzielenia linii

Funkcje formatowania ciągów
Funkcja wordwrap()

W mądrość Jasia wierząc skrycie, Pytał chłopca nauczyciel O powstańców polskich liczne Wpływy ideologiczne. Cisza. Rozpacz. Pustka w głowie. Więcej zgoła nic, albowiem Pozbawionym wszelkiej wiedzy Nie pomogą i koledzy.

użycie funkcji wordwrap($ciagznakow, 20, "*\n")

W mądrość Jasia* wierząc* skrycie, Pytał* chłopca* nauczyciel O* powstańców* polskich* liczne Wpływy* ideologiczne. * Cisza. Rozpacz.* Pustka w głowie. * Więcej zgoła* nic, albowiem * Pozbawionym* wszelkiej wiedzy * Nie pomogą i* koledzy.

użycie funkcji wordwrap() z funkcją nl2br()

W mądrość Jasia* wierząc* skrycie, Pytał* chłopca* nauczyciel O* powstańców*

```
19  echo $ciagznakow.'<br>';
20  echo '<h2>użycie funkcji wordwrap($ciagznakow, 20, "*\n") </h2>';
21  echo wordwrap($ciagznakow, 20, "*\n");
22  echo '<h2>użycie funkcji wordwrap() z funkcją nl2br() </h2>';
23  echo nl2br(wordwrap($ciagznakow, 20, "*\n"));
24  echo '<h2>użycie funkcji wordwrap() jeżeli wyraz jest za długi</h2>';
25  echo wordwrap('bardzodługiwyraz', 10, "*\n", true);
```

bardzodłu* giwyraz

Funkcje formatowania ciągów
Funkcja wordwrap()

użycie funkcji wordwrap() z argumentami

W mądrość Jasia
wierząc
skrycie, Pytał
chłopca
nauczyciel O
powstańców
polskich
liczne Wpływy

```
27  echo '<h2>użycie funkcji wordwrap() z argumentami</h2>';
28  echo wordwrap($ciagznakow, 20, "<br>\n");
```

Funkcje zmiany wielkości liter

strtoupper($str) – zamiana liter na duże
strtolower($str) - zamiana liter na małe

```php
<!DOCTYPE html>
<html lang="pl">
<head>
<title>funkcje zmiany wielkości liter()</title>
<meta charset="utf-8">
</head>
<body>
<?php
$a="ęóąśłżźćń";
$b="ĘÓĄŚŁŻŹĆŃ";
echo strtoupper($a);
echo "<br />";
echo strtolower($b);
?>
</body>
</html>
```

ęóąśłżźćń
ĘÓĄŚŁŻŹĆŃ

Funkcje zmiany wielkości liter

PHP - funkcja strtolower/strtoupper i problemy z polskimi znakami(kodowanie)

Znane funkcje PHP (strtolower(), strtoupper()) mogą sprawiać problemy z polskimi znakami (zwłaszcza w przypadku popularnego kodowania UTF-8).

Jeżeli korzystając z tych funkcji otrzymujemy niepokojące wyniki, należy zastosować:

```
1 string mb_strtolower($str,[$encoding])
2 string mb_strtoupper($str,[$encoding])
```

np.

```
1 echo mb_strtoupper($a,"UTF-8");
2 echo "<br />";
3 echo mb_strtolower("ąęłśćń","UTF-8");
```

Teraz otrzymamy poprawny wynik.

Funkcje zmiany wielkości liter

ucfirst($str) – pierwsza litera ciągu zamieniona na dużą literę
ucwords($str) – pierwsze litery wyrazów w ciągu są zamienione na dużą literę

```php
<?php
$ciagznakow='w mądrość Jasia wierząc skrycie,
pytał chłopca nauczyciel
o powstańców polskich liczne
wpływy ideologiczne.';
echo '<h2>użycie funkcji ucfirst() </h2>';
echo ucfirst($ciagznakow).'<br>';
echo '<h2>użycie funkcji ucwords() </h2>';
echo ucwords($ciagznakow).'<br>';
?>
```

użycie funkcji ucfirst()

W mądrość Jasia wierząc skrycie, pytał chłopca nauczyciel o powstańców polskich liczne wpływy ideologiczne.

użycie funkcji ucwords()

W Mądrość Jasia Wierząc Skrycie, Pytał Chłopca Nauczyciel O Powstańców Polskich Liczne Wpływy Ideologiczne.

Funkcje usuwania ciągu znaków

Do usunięcia z początku lub końca ciągu białych znaków
{ '\n' (symbol nowego wiersza), '\r' (symbol powrotu karetki), '\t' (poziomy tabulator), '\x0B' (pionowy tabulator), '\0' (znak końca ciągu) i znak spacji }
można użyć jednej z trzech funkcji

trim($str) – usuwa podane znaki z początku i z końca ciągu
ltrim($str) – usuwa podane znaki z początku ciągu
rtrim($str) – usuwa podane znaki z końca ciągu

Do usuwania innych znaków należy użyć powyższych funkcji z dodatkowym parametrem (w takim przypadku możliwe jest podane zakresu znaków, oddzielając znaki skrajne dwiema kropkami, np.: 'a..f')
postaci:
nazwa_funkcji($str, 'znaki do usunięcia');

```
$nazwisko = trim($_POST['nazwisko']);
$adres    = trim($_POST['adres']);
$mail     = trim($_POST['mail']);
```

Długość łańcucha znaków

Długość ciągu znaków zwraca funkcja strlen().

```
8  <?php
9
10   $str    = "Krótki łańcuch";
11   echo $str .'<br>';
12   $n=strlen($str);
13   echo 'dlugosc ciagu znaków: '.$n.'<br>';
14
15 for($i=0;$i<$n;$i++)
16     echo($str[$i]).'<br>';
17 ?>
```

Formatowanie ciągów do przechowania w bazie danych

Niektóre znaki, będące częścią łańcuchów znakowych, mogą sprawiać kłopoty przy dopisywaniu do baz danych. Znakami takimi są przede wszystkim cudzysłowy (pojedynczy i podwójny), lewy ukośnik oraz symbol NULL.

W takim przypadku konieczne jest znalezienie sposobu na zaznaczenie tych znaków, czyli na ucieczkę od nich. Do ucieczki od tych znaków służy znak lewego ukośnika. Przy użyciu tego znaku, np. cudzysłów (") zostaje zapisany jako \", sam ukośnik (\) staje się podwójnym ukośnikiem \\, itd.

W PHP istnieją dwie specjalne funkcje, służące do ucieczki od znaków specjalnych. I tak, przed zapisem jakiegokolwiek łańcucha znakowego do bazy danych należy go przeformatować przy użyciu funkcji addSlashes(). Funkcja ta pobiera łańcuch znaków, jako argument i zwraca ten łańcuch, ale przeformatowany.

Formatowanie ciągów do przechowania w bazie danych

```
 <?
  $comment = addslashes(trim($_POST['comment']));
?>
```

Zastosowanie funkcji 'addSlashes()' spowoduje, że jeżeli jest taka potrzeba, ciąg zostanie uzupełniony o stosowne ukośniki.

Po odczytaniu łańcucha znaków z bazy danych, należy przwrócić oryginalną składnię znaków specjalnych. Do tego celu służy funkcja 'stripslashes()'.

```
<?
  $comment = stripslashes($comment);
?>
```

Konfiguracja PHP na serwerze może być tak ustawiona, że interpreter automatycznie umieszcza i usuwa ukośniki. Można się o tym przekonać wywołując funkcję get_magic_quotes_gpc(), która w takim przypadku zwróci wartośc true. Przy włączeniu tej dyrektywy, by uniknąc wyświetlania ukośników, dane użytkownika można wyświetlać po wywołaniu funkcji stripslashes().

Rozdzielanie i łączenie całego łańcucha

array explode(separator,string,limit)

string implode(separator,array)

```php
<?php
// string do rozdzielenia
$dane = "Jan,Kowalski,Bielsko-Biała";

// wykorzystanie funkcji explode, wg przecinka
$dane_osobowe = explode(",", $dane);

// wyświetlenie otrzymanej tablicy
echo $dane_osobowe[0]."<br/>";
echo $dane_osobowe[1]."<br/>";
echo $dane_osobowe[2]."<br/>";
$tekst = "Litwo, ojczyzno moja, Ty jesteś jak zdrowie.";
$wyrazy = explode(" ", $tekst);
// wyświetli ilość wyrazów w zmiennej $tekst
echo count($wyrazy);
//poniżej tablica
$tab=array('element1', 5, 'abcde', '6');
//elementy tablicy połączone znakiem *
$tabStr=implode('*',$tab);
echo '<br>';
echo $tabStr;
?>
```

Jan
Kowalski
Bielsko-Biała
7
element1*5*abcde*6

Znajdowanie podciągów

Podstawową funkcją wyszukiwania jednego ciągu w drugim jest strpos() posiadająca także wariant stripos(), w którym nie gra roli wielkość liter.

Funkcja strpos() - przyjmuje ona dwa argumenty, ciąg szukany oraz ciąg, w którym szukamy. W przypadku znalezienia wyrazu wewnątrz szukanego stringa, funkcja zwróci pozycję występowania wyrazu. Jeżeli natomiast podciąg nie zostanie znaleziony, funkcja zwróci wartość logiczną FALSE. Funkcja przyjmuje opcjonalnie trzeci paramatr, w którym można ustawić numer znaku, od którego funkcja ma zacząć przeszukiwać.

```
 8  <?php
 9    $text    = "Witaj, świecie!";
10    //szukanie spacji
11    if(strpos($text,' ') ) echo 'Łańcuch zawiera spacje <br >';
12    //szukanie 'świecie'
13    echo "Łańcuch zawiera podciąg -świecie- na pozycji nr ". strpos($text,
      "świecie").'<br >';
14  ?>
```

Łańcuch zawiera spację
Łańcuch zawiera podciąg -świecie- na pozycji nr 7

Znajdowanie podciągów

```
substr(string,start,length)
```

Parameter	Description
string	Required. Specifies the string to return a part of
start	Required. Specifies where to start in the string • A positive number - Start at a specified position in the string • A negative number - Start at a specified position from the end of the string • 0 - Start at the first character in string
length	Optional. Specifies the length of the returned string. Default is to the end of the string. • A positive number - The length to be returned from the start parameter • Negative number - The length to be returned from the end of the string

Znajdowanie podciągów

Funkcja strstr() służy do sprawdzania, czy podany ciąg jest fragmentem innego ciągu. Funkcja przyjmuje dwa parametry: przeszukiwany ciąg ($str1) i szukany ciąg ($str2). Jeśli dany ciąg jest fragmentem podanego, to zwracany jest ciąg – fragment przeszukiwanego ciągu od pierwszego wystąpienia szukanego ciągu do jego końca.

```
10  $email = "prezydent@polska.pl";
11  $domena = strstr($email, "@");
12  echo $domena.'<br>';
13  if(strstr($email, "polska")!==False)
14     echo "Email ma w sobie słowo 'polska'";
15
```

@polska.pl
Email ma w sobie słowo 'polska'

Dołączanie plików

Jeżeli skrypt PHP zawiera dużą ilość kodu, to można go podzielić i zapisać w kilku oddzielnych plikach.
Do ponownego ich połączenia można użyć instrukcji: include('nazwa_pliku') require('nazwa_pliku').
Można użyć również include 'nazwa_pliku' require 'nazwa_pliku'.

Obydwie instrukcje w miejscu wystąpienia wstawiają zawartość wskazanego pliku.
Dołączony plik zostanie wykonany tak, jakby był częścią kodu, w którym został wstawiony.

Różnica między funkcjami występuje tylko wtedy, gdy dołączony plik nie może zostać odczytany. Instrukcja include wygeneruje ostrzeżenie działania nie, ale skrypt zawierający jej wykonanie będzie nadal działał. Natomiast użycie instrukcji require spowoduje zgłoszenie błędu i zakończenie działania skryptu

funkcja header() w PHP

header -- Wysyła surowy nagłowek HTTP

HTTP (ang. Hypertext Transfer Protocol)to protokół internetowy, za pomocą którego przesyłane są do serwera żądania udostępnienia dokumentów WWW, a także dane wprowadzone do formularzy oraz zwrotnie zawartości stron WWW i inne pokrewne dane (np. dokumenty XML).

Modyfikując zapytania przesyłane do serwera, można kontrolować przechowywania strony w cache'u przeglądarki, zmieniać kody odpowiedzi, dynamicznie przekierowywać zapytania, a także przesyłać dane metodą POST, nie korzystając przy tym z formularzy.

void **header** (string treść_nagłówka [, bool zamień [, int kod_odpowiedzi_http]])

Argument opcjonalny „zamień” określa, czy funkcja ma zastąpić nagłówek tego samego typu przygotowany przez serwer, czy dodać jeszcze jeden. Domyślnie, oryginalny nagłówek zostanie zastąpiony, ale jeśli ustawimy ten argument na FALSE, to nowy nagłówek zostanie dodany do już istniejących.
Drugi argument opcjonalny „kod_odpowiedzi_http” pozwala narzucić określony kod odpowiedzi HTTP. (Argument ten jest dostępny w PHP 4.3.0 i wyższych).

Przykłady stosowania funkcji header()

1) funkcja header wyśle do przegladarki wskazany plik
header('location: jakisplik');

Ponieważ funkcja header nie kończy działania skryptu, należy użyć funkcji exit().
Istnieje także inna funkcja, o podobnym działaniu jak exit(). Jest to funkcja die()

```
7  <body>
8  <a href="pobierz.php">tekst.rar</a> (pobrań: <?=filesize("licznik.txt") ; ?>)
9  </body>
```

```
1  <?php
2  $f = "licznik.txt";
3  $file = fopen($f, 'a');
4  fputs($file, ".");
5  fclose($file);
6  header("location: tekst.rar");
7  exit();
8  ?>
```

Każde uruchomienie tego skryptu, wywołane po kliknięciu na link, doda kropkę do naszego licznika. Liczba kropek to objętość pliku odczytywana przez funkcję filesize(), a więc nasz licznik kliknięć. Funkcja header() wyśle do przeglądarki wskazany plik tekst.zip tak, jakby użytkownik na niego kliknął.

Przykłady stosowania funkcji header()

Nalezy pamiętać, aby przed wywołaniem funkcji header nie wysłać do przeglądarki żadnego tekstu!

1) przechodzenie do innej strony

```
header('location: jakisSkrypt.php');
```

Ponieważ funkcja header nie kończy działania skryptu, należy użyć funkcji exit().
Istnieje także inna funkcja, o podobnym działaniu jak exit(). Jest to funkcja die()

2) Ustawianie kodowania

```
header('Content-type: utf-8');
```

funkcja header() w PHP