PHP

PHP – Wprowadzenie

- Akronim PHP (*ang. Personal Home Page*)
- Tworzonym na zasadach Open Source
- Dostępny
  - jako moduł lub CGI
  - pod wieloma platformami: Unix, Linux, Windows, MacOS
  - z wieloma serwerami HTTP: Apache, IIS, iPlanet, Xitami, OmniHTTPd

PHP – Wprowadzenie

- Język skryptowy uruchamiany po stronie serwera
- Możliwość zagnieżdżania znaczników HTML
- PHP może generować dowolny kod, czyli poza HTML np. XML lub nawet obrazki

- Strony ze skryptami:
- – Pliki z rozszerzeniem **.php**
- – Wymagane prawo do odczytu pliku (nie CGI)
- – Składnia podobna do języka C++
- – Wyświetlane dane stają się częścią strony

Historia

- • 1995 – PHP/FI (skrypty Perl)
- (Personal Home Page / Forms Interpreter)
- • 1997 – PHP/FI 2.0
- • 1998 – PHP 3.0
- • 2000 – PHP 4.0
- • 2004 – PHP 5.0

PHP – Pierwszy przykład

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
 <title>Przykładzik</title>
</head>
<body>
 <?php echo "Oto przykład pierwszy."; ?>
</body>
</html>
```

PHP – sposoby użycia

- Skrypty po stronie serwera
  - serwer WWW
  - parser PHP
  - przeglądarka
- Skrypty z linii poleceń
  - parser PHP
  - zastosowanie do przetwarzania tekstu
  - uruchamiane w cronie lub menedżerze zadań

PHP i bazy danych

- Adabas D
- Ingres
- Oracle (OCI7 i OCI8)
- dBase
- InterBase
- Ovrimos
- Empress
- FrontBase
- PostgreSQL
- FilePro
- mSQL
- SolidHyperwave
- Direct MS-SQL
- Sybase
- IBM DB2
- MySQL
- Velocis
- Informix
- ODBC
- Unix dbm

PHP – osadzanie skryptów

```
<? echo "To jest skrypt"; ?>

<?= "To jest ten sam skrypt"; ?>

<?php echo "W XHTML i XML lepiej tak"; ?>

<% echo "To znowu jest skrypt"; %>

<%= "To znowu jest ten sam skrypt"; %>

<script language="php">
  echo "Można też tak";
</script>
```

PHP – kilka uwag

- Instrukcje należy kończyć średnikiem
- PHP poza trybem "PHP" przepisuje tekst bez zmian
- Komentarze
  - `//, #` – jednowierszowe
  - `/* ... */` – blokowe

PHP – zmienne

- Zmienne są dynamicznie typowane
- Poprzedzamy znakiem $
- W nazwach małe i wielkie litery są rozróżniane

```
<?
$Imie = "Jas";
$imie = " Marysia";
$Nazwisko = " Ararat";
echo $Imie, $imie, $Nazwisko;
?>
```

PHP – zmienne

- Przypisanie przez referencję:

```
<?
   $halasuje = "Zmyłer";
   $sondaze = &$halasuje;
   $sondaze = „Gaduła";
   echo $halasuje." i prowadzi ".$sondaze;
?>
```

- Tylko nazwane wyrażenia mogą być przypisane przez referencję

```
<? $heja = &(666*69); ?> // ŹLE
```

PHP – funkcje do zmiennych

- get_defined_vars
- get_resource_type
- doubleval
- floatval
- strval
- is_array
- is_bool
- is_callable
- is_double
- is_float
- is_int
- is_integer
- is_long
- is_numeric
- is_object
- is_real
- is_resource
- is_scalar
- is_string
- is_null
- print_r
- settype
- serialize
- unserialize
- isset
- unset

PHP – stałe

- Definiujemy za pomocą funkcji **`define()`**

```
bool define(string nazwa,
                mixed wartość
            [, bool case_insensitive])
```

```
<?
  define(STALA,"Ahoj!!!");
  echo Stala; // Stala
  define(STALA,"Ahoj!!!",true);
  echo Stala; // Ahoj!!!
?>
```

PHP – stałe

- Sprawdzamy czy zdefiniowana za pomocą funkcji **`defined()`**

`bool defined (string nazwa_stalej)`

```
<?
  define(STALA,"Ahoj!!!",true);
  echo(defined('Stala') ? "Jest" : "Nie jest");
?>
```

PHP – stałe

- Funkcja `mixed constant(string name)` pobiera wartość stałej
- Funkcja `array get_defined_constants()` pobiera tablicę stałych

```
<?
  print_r(get_defined_constants());
?>
```

PHP – typy danych

- Skalarne
  - boolowski: boolean
  - liczby całkowite: integer
  - liczby zmiennoprzecinkowe: float
  - łańcuchy znaków: string

PHP – typy danych

- Złożone
  - tablice: array
  - obiekty: object
- Specjalne
  - identyfikatory zasobów: resource
  - puste: null

PHP – typy danych

- Boolowski: **`TRUE,FALSE`**
- Liczby całkowite: **`1234,-123,0123,0xABC`**
- Liczby zmiennoprzecinkowe: **`1.2,3E-2`**
- Łańcuchy znaków
  - pojedyncze cudzysłowy
  - podwójne cudzysłowy
  - składnia heredoc

PHP – więcej o napisach

- Podwójne cudzysłowy powoduje zamianę nazwy zmiennej na jej wartość
- Znaki specjalne: `\n, \r, \t, \\, \$, \"`
- Składnia heredoc

```
$s = <<<EOD
To jest taki sobie\n
napis żeby było
EOD;
```

- Offset: `$s{0}, $s[5]`

PHP – przykłady

- `echo 'Tekst'`
- `echo 'To jest \'cytat\' stulecia'`
- `echo 'Taki "\'c\'u\'ś\'" inny'`

- `$zmienna = "papier"`
- `echo "$zmienna"`
- `echo "${zmienna}owe torebki"`

PHP – tablice

- Konstruktor **`array`**

  `array( [index =>] wartość, ...)`
  - index – napis lub nieujemna liczba całkowita
  - wartość – cokolwiek
- Przypisywanie wartości
  - **`$tablica[index] = wartość;`**
  - **`$arr[] = wartość;`**

    // ideksem będzie kolejna liczba

PHP – tablice

- Usuwanie pary (index, wartość)

```
$a = array(1 => 'jeden',
           2 => 'dwa',
           3 => 'trzy' );
unset($a[2]);
/* to tak, jakby utworzyć tablicę:
$a = array(1 => 'jeden', 3 => 'trzy');
   ale NIE:
$a = array(1 => 'jeden', 2 => 'trzy');
*/
```

PHP – tablice

```
$wypasiona_zmienna = array(
 "owoce" =>
   array("a" => "pomarańcza",
         "b" => "banan",
         "c" => "jabłko"),
 "liczby" => array(1,2,3,4,5,6),
 "dziury" => array(
        "pierwsza",
   5 => "druga",
        "trzecia")
);
$x = $wypasiona_zmienna['dziury'][5];
```

PHP – obiekty

```
<?php
  class kasa{
    function rabuj()
    {
        echo "Rabujemy kase...";
     }
}
$bzwbk = new kasa();
$bzwbk->rabuj();
?>
```

PHP – if, else, elseif

```
<?php
if ($var > 5)
{
  $answ = $var1;
}
else
{
  $answ = $var2;
}
?>
```

```
$answ = ( $var > 5 ? $var1 : $var2 );
```

PHP – if, else, elseif

```
<?php
if (wyrazenie-logiczne) {
?>
<strong>prawda</strong>
<?php
} else {
?>
<strong>fałsz</strong>
<?php
}
?>
```

PHP – if, else, elseif

```
if ($a > $b) {
  print "a jest większe niż b";
} elseif ($a == $b) {
  print "a jest równe b";
} else {
  print "a jest mniejsze niż b";
}
```

```
<?php if ($a == 5): ?>
A jest równe 5
<?php endif; ?>
```

PHP – pętla while..do

```
$i = 1;
while ($i <= 10) {
  print $i++;
}
```

```
$i = 1;
while ($i <= 10):
  print $i;
  $i++;
endwhile;
```

PHP – pętla do..while

```
$i = 0;
do {
  print $i;
} while ($i>0);
```

PHP – pętla for

- ```
  for ($i = 1; $i <= 10; $i++) {
    print $i;
  }
  ```
- ```
  for ($i = 1;;$i++) {
    if ($i > 10) break;
    print $i;
  }
  ```
- ```
  $i = 1;
  for (;;) {
      if ($i > 10) break;
      print $i++;
  }
  ```
- ```
  for ($i = 1; $i <= 10; print $i, $i++);
  ```

PHP – break, continue

- Break i continue akceptuje opcjonalny argument, który mówi, ile zagnieżdżonych struktur kontrolnych ma zostać opuszczonych w danym momencie

```
$i=0;
while (++$i) {
  for ($j=0; $j<10; $j++)
  if ($i*$j==12) break 1;
  if ($i*$j==69) break 2;
}
```

PHP – switch

```
<?
switch ($i):
  case 0:
    print "i jest równe 0"; break;
  case 1:
    print "i jest równe 1"; break;
  default:
    print "i jest różne od 0 i 1";
endswitch;
?>
```

PHP – switch

```
<?
$imie = 'ala';
switch ($imie) {
  case 'ala':
    echo "ma kota"; break;
  case 'milicja':
    echo "ma alę"; break;
  default: echo "reszta się nie liczy";
}
?>
```

PHP – return, ...

- Instrukcja return(): wywołana z wnętrza funkcji, natychmiastowo kończy wykonywanie tej funkcji i zwraca jako jej wartość swój argument.
- Składnia alternatywna: dostępna dla poleceń if, while, for, foreach i switch. Polega na zamianie nawiasu otwierającego { na dwukropek, a nawiasu zamykającego } na słowo endif, endwhile, endfor, endforeach i endswitch odpowiednio.

PHP – zewnętrzne pliki

- Instrukcje
  - include()
  - require()
  - include_once()
  - require_once()
- Czym to się różni?

PHP – zewnętrzne pliki

vars.php

```
<?php
  $kolor = 'zielone';
  $owoc = 'jabłko';
?>
```

test.php

```
<?php
  echo "Jedno $kolor $owoc"; // Jedno
  include 'vars.php';
  echo "Jedno $kolor $owoc";
    // Jedno zielone jabłko
?>
```

PHP – obsługa napisów

- void echo string s1 [, string sn ...]
  - wypisuje s1, ..., sn
  - echo "ala","ma kota"
- string ltrim(string str [, string charlist])
  - usuwa białe znaki z początku str; opcjonalnie można podać w charlist jakie znaki ma usunąć
  - domyślnie usuwa: " ", "\t", "\n", "\r", "\0", "x0B" (ASCII: 32, 9, 10, 13, 0, 13)

PHP – obsługa napisów

- string rtrim(string str [, string charlist])
  - usuwa białe znaki z końca str
- string trim(string str [, string charlist])
  - usuwa białe znaki z obu stron str
- string chr(int ascii)
  - zwraca string zawierający znak o kodzie określonym w ascii
- int ord(string str)
  - zwraca kod pierwszego znaku z str

PHP – obsługa napisów

- int strcmp(string str1, string str2)
  - jeśli str1<str2, to zwraca -1
  - jeśli str1>str2, to zwraca 1
  - jeśli str1=str2, to zwraca 0
- int strpos(string gdzie, string co [, int offset])
  - wyszukuje pierwszego wystąpienia co w gdzie
  - jeśli co nie występuje w gdzie zwracane jest coś równoważne FALSE
  - jeśli podamy offset, to wyszukiwanie rozpocznie się od pozycji określonej przez offset.

PHP – obsługa napisów

- int strrpos(string gdzie, char co)
  - działa podobnie do strpos() tylko, że wyszukuje od prawej strony
  - możemy wyszukiwać tylko pojedynczy znak
- int strlen(string s)
  - zwraca długość s
- string strtolower(string str)
  - zwraca str z powymienianymi wielkimi literami na małe

PHP – obsługa napisów

- string strtoupper(string str)
  - zwraca str z powymienianymi małymi literami na wielkie
- string strstr(string str1, string str2)
  - zwraca część str1 od pierwszego wystąpienia str2

```
$email = 'pawel@wp.pl';
$domena = strstr($email, '@');
echo $domena; // @wp.pl
```

PHP – obsługa napisów

- string substr(string str, int start [, int length])
  - zwraca length znaków str od pozycji start;
  - domyślnie length jest do końca str

```
echo substr("abcdef", 1);     // "bcdef"
echo substr("abcdef", 0, 4); // "abcd"
echo substr("abcdef", 0, 8); // "abcdef"
```

- int substr_count(string str1, string str2)
  - zwraca ilość wystąpień str2 w str1

```
echo substr_count("aaaaa","aa") // 2
```

PHP – obsługa napisów

- int substr_replace(string s1, string s2,
  int start [, int length])
  - wymienia w s1 część długości length i zaczynającą się w start na s2

```
echo substr_replace("Łosie mają w
nosie","grają na",6,6);
// Łosie grają na nosie
```

PHP – obsługa napisów

- array explode(string separator, string str
  [, int limit])
  - dzieli str względem separator
  - jeśli ustawimy limit, to tablica wynikowa będzie zawierała co najwyżej limit elementów, w ostatnim resztę str

```
$liczby = "1,23,32,44";
$tablica = explode(",", $liczby);
echo $tablica[1]; // 23
```

PHP – obsługa napisów

- string implode(string glue, array pieces)
  - tworzy string zawierający po kolei elementy z pieces rozdzielone glue

```
$tab = array('osoba','email','tel');
$lista = implode(",", $array);
echo $lista; // osoba,email,tel
```

- string strrev(string str)
  - odwraca str

```
echo strrev("Kilof"); // "foliK"
```

PHP – obsługa napisów

- string strip_tags(string str [, string dobre_tagi])
  - usuwa z str tagi HTML poza tymi określonymi w dobre tagi

```
$s = strip_tags($string,'<a><b><i><u>')
```

- string wordwrap (string str [, int width [, string break [, boolean cut]]])
  - łamie str w białych znakach do długości width, słowem break
  - jeśli chcemy przecinać za długie słowa, to cut ustawiamy na 1

PHP – obsługa napisów

- funkcja wordwrap() – przykład

```
$text = "Bardzo długie i nie za fajne
  słooooooooooooooooooowo";
$goodtekst = wordwrap($text,8,"<br>\n");
echo "$goodtext"

Bardzo
długie i
nie za
fajne
słooooooooooooooooooowo
```

PHP – obsługa napisów

- funkcja wordwrap() – drugi przykład

```
$goodtext = wordwrap($text,8,"<br>\n",1);
echo "$goodtext"
```

```
Bardzo
długie i
nie za
fajne
słooooooo
oooooooo
ooooooowo
```

PHP – obsługa napisów

- string addslashes (string str)
  - dodaje znak "\" tam gdzie to jest potrzebne

```
$str = "Is your name O'reilly?";
// Is your name O\'reilly?
echo addslashes($str);
```

- string addcslashes (string str, string charlist)
  - dodaje znak "\" przed każdy znak z charlist

```
$text = "Ala ma kota";
echo addcslashes($text, "ma");
// Al\a \m\a kot\a
```

PHP – obsługa napisów

- string nl2br(string str)
  - zwraca str z wstawionymi **`<br />`** przed każdym "\n"
- string htmlspecialchars (string string [, int quote_style [, string charset]])
  - wymienia znaki: **`<`**, **`>`**, **`&`**, **`"`**, **`'`** na ich kody: **`<`** **`>`** **`&`** **`"`** **`'`**
  - ' wymienia, jeśli włączone ENT_QUOTES
  - " wymienia, jeśli wyłączone ENT_NOQUOTES

PHP – obsługa napisów

- string md5(string str)
  - oblicza MD5 hash od str
- string md5_file (string filename
  [, bool raw_output])
  - oblicza MD5 hash pliku podanego jako parametr
- string bin2hex (string str)
  - zamienia ciąg binarny na heksadecymalną reprezentację