

LEKSYKON  
KIESZONKOWY

# PHP 5

Praktyczna pomoc na co dzień  
— sięgnij po świetną ściągawkę z PHP!

Struktura języka,  
czyli znaczniki, typy danych,  
operatory i inne elementy

Instrukcje sterujące oraz funkcje,  
czyli wydawanie różnych poleceń

Programowanie obiektowe,  
czyli najkrótsza droga do celu

MARCIN LIS

Helion



## » Idź do

- Spis treści
- Przykładowy rozdział
- Skorowidz

## » Katalog książek

- Katalog online
- Zamów drukowany katalog

## » Twój koszyk

- Dodaj do koszyka

## » Cennik i informacje

- Zamów informacje o nowościach
- Zamów cennik

## » Czytelnia

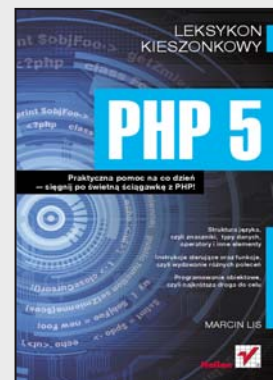
- Fragmenty książek online

## » Kontakt

Helion SA  
ul. Kościuszki 1c  
44-100 Gliwice  
tel. 32 230 98 63  
e-mail: helion@helion.pl  
© Helion 1991–2011

## PHP 5. Leksykon kieszonkowy

Autor: [Marcin Lis](#)  
ISBN: 978-83-246-3394-4  
Format: 122×194, stron: 168



### Praktyczna pomoc na co dzień – sięgnij po świetną ściągawkę z PHP!

- Struktura języka, czyli znaczniki, typy danych, operatory i inne elementy
- Instrukcje sterujące oraz funkcje, czyli wydawanie różnych poleceń
- Programowanie obiektowe, czyli najkrótsza droga do celu

Język skryptowy PHP od lat wciąż na nowo udowadnia swoją elastyczność i niezawodność. Wykorzystywany do tworzenia dynamicznych serwisów internetowych, a także samodzielnych aplikacji, szczególnie dobrze sprawdza się wtedy, gdy chcemy zapewnić możliwość współdziałania naszej witryny z różnymi formatami danych oraz skomplikowanymi bazami. Ponadto jego opanowanie nie nastęrcza wielu trudności, a efekty pracy nawet początkujących webmasterów bywają wręcz spektakularne. Jeśli masz już za sobą pierwsze kroki w dziedzinie programowania w PHP5, w codziennej pracy z pewnością przyda Ci się poręczna ściągawka, w której zawarto najważniejsze informacje i sposoby wykonywania podstawowych zadań programistycznych. Właśnie trzymasz ją w ręku!

„PHP5. Leksykon kieszonkowy” powstał po to, by ułatwić Ci szybkie przypomnienie sobie wybranych zagadnień związanych z tą tematyką. Znajdziesz tu informacje o instalacji oraz konfiguracji środowiska w systemach Linux i Windows, strukturze języka, instrukcjach sterujących czy funkcjach. Przeczytasz o operacjach na danych, a także o obsłudze ciągu znaków daty i czasu. Przypomnisz sobie, jak używać tablic, współpracować z serwerem i przeglądarką, obsługiwać wyjątki. W leksykonie odszukasz także wiadomości dotyczące systemu plików i programowania obiektowego. Wszystko to wydatnie skróci Twoją pracę i pomoże Ci uniknąć wielu potencjalnych błędów.

- Instalacja i konfiguracja
- Struktura języka
- Instrukcje sterujące
- Funkcje
- Operacje na danych i obsługa ciągu znaków
- Tablice
- Obsługa daty i czasu
- Współpraca z serwerem i przeglądarką
- Programowanie obiektowe
- Wyjątki
- Obsługa systemu plików

**Dobra podpowiedź zawsze pod ręką!**

<b>Wstęp .....</b>	<b>7</b>
<b>1. Instalacja i konfiguracja .....</b>	<b>9</b>
Linux	9
Windows	11
Plik konfiguracyjny php.ini	13
PHP w wierszu poleceń	14
<b>2. Struktura języka .....</b>	<b>15</b>
Struktura skryptów	15
Znaczniki skryptów PHP	15
Znaczniki kanoniczne	16
Znaczniki skryptów HTML	16
Znaczniki typu SGML	16
Znaczniki typu ASP	16
Osadzanie skryptów w kodzie (X)HTML	17
Przeplatanie kodu (X)HTML i PHP	18
Komentowanie kodu	18
Typy danych	19
Literały	22
Identyfikatory	25
Słowa kluczowe (zarezerwowane)	25
Zmienne	26
Zmienne nazwy zmiennych	27
Zmienne superglobalne	28
Automatyczne i wymuszone konwersje typów	30
Stałe	36
Operatory	40
Przetwarzanie wyrażeń	54

<b>3. Instrukcje sterujące .....</b>	<b>57</b>
Instrukcje warunkowe	57
Pętle	59
Instrukcje break i continue	62
Instrukcja goto	65
Składnia alternatywna	66
Łączenie skryptów	69
<b>4. Funkcje .....</b>	<b>71</b>
Deklaracje	71
Zwracanie wartości	71
Wywoływanie funkcji	72
Sposoby przekazywania argumentów	72
Zasięg zmiennych	75
Wartość zmiennej jako nazwa funkcji	77
Definiowanie funkcji wewnątrz funkcji	78
Funkcje anonimowe	78
<b>5. Wyprowadzanie danych i obsługa ciągów znaków .....</b>	<b>80</b>
Obsługa standardowego wyjścia	80
Wyprowadzanie znaczników HTML	81
Różnice między typami ciągów znaków	81
Znaki specjalne	82
Dostęp do pojedynczych znaków ciągu	83
Funkcje przetwarzające ciągi znaków	83
<b>6. Tablice .....</b>	<b>104</b>
Tablice indeksowane numerycznie	104
Tablice asocjacyjne	105
Tablice wielowymiarowe	106
Rozmiar tablicy	107
Odczyt danych za pomocą pętli	108
Funkcje wspomagające przetwarzanie tablic	109
<b>7. Obsługa daty i czasu .....</b>	<b>116</b>
Funkcja checkdate (dostępna od PHP 3)	116
Funkcja date (dostępna od PHP 3)	116

<b>8. Współpraca z serwerem i przeglądarką .....</b>	<b>125</b>
Odbieranie danych wysłanych metodą GET	125
Odbieranie danych wysłanych metodą POST	126
Wysyłanie plików do serwera WWW	126
Odbiór plików przesłanych do serwera	127
Obsługa cookies	129
Obsługa sesji	131
Opcje konfiguracyjne sesji	133
<b>9. Programowanie obiektowe .....</b>	<b>136</b>
Definiowanie klasy	136
Składowe klasy	136
Specyfikatory dostępu	137
Tworzenie obiektów	137
Odwołania do składowych	138
Konstruktory i destruktory	138
Przeciążanie metod	140
Dziedziczenie	141
Przesłanianie składowych	142
Metody finalne	142
Klasy finalne	143
Składowe statyczne	143
<b>10. Wyjątki .....</b>	<b>145</b>
Zgłaszanie wyjątków	145
Klasa Exception	145
Sekcja try...catch	146
Wyjątki predefiniowane	147
Tworzenie wyjątków	147
<b>11. Obsługa systemu plików .....</b>	<b>148</b>
Odczyt zawartości katalogu	148
Tworzenie i usuwanie katalogów	149
Tworzenie i usuwanie plików	150
Otwieranie i zamykanie plików	150
Odczyt plików	152
Zapis danych w plikach	153
<b>Skorowidz .....</b>	<b>157</b>

# Rozdział 4. Funkcje

## Deklaracje

W celu utworzenia funkcji należy użyć słowa kluczowego `function`. Taka konstrukcja ma schematyczną postać:

```
function nazwa_funkcji()  
{  
    //instrukcje wnętrza funkcji  
}
```

Przy nazywaniu funkcji obowiązują takie same zasady jak przy innych identyfikatorach. Jeżeli funkcja ma przyjmować argumenty, ich listę należy umieścić w nawiasie okrągłym występującym za nazwą:

```
function nazwa_funkcji($argument1, $argument2,...,$argumentN)  
{  
    //instrukcje wnętrza funkcji  
}
```

W ciele (we wnętrzu) funkcji można zawrzeć dowolny, poprawny składniowo, kod PHP, włącznie z definicjami innych funkcji i klas.

Oto przykład funkcji wyświetlającej napis:

```
function wyświetl()  
{  
    echo "Tekst generowany przez funkcję wyświetl.";  
}
```

## Zwracanie wartości

Zwracanie wartości przez funkcję jest realizowane za pomocą instrukcji `return`. Jeśli wystąpi ona wewnątrz funkcji, ta jest przerywana i zwraca wartość występującą po `return`. Schematycznie tego typu konstrukcja wygląda następująco:

```
function nazwa_funkcji(argumenty)  
{  
    //instrukcje wnętrza funkcji  
    return wartość;  
}
```

W miejscu wywołania takiej funkcji zostanie wstawiona zwrócona przez nią wartość, która będzie mogła być wykorzystana w dalszej części

skryptu. Jeżeli zostanie użyta instrukcja `return` bez wskazanej wartości, funkcja jest przerywana i zwracana wartość `null`. Jeśli funkcja nie zawiera instrukcji `return`, zwracaną wartością jest również `null`.

Oto przykład funkcji zwracającej wartość arytmetyczną całkowitoliczbową (`integer`) 12:

```
function func()
{
    return 12;
}
```

A to przykład funkcji przyjmującej dwa argumenty i zwracającej wynik ich dodawania:

```
function dodaj($arg1, $arg2)
{
    return $arg1 + $arg2;
}
```

## Wywoływanie funkcji

Aby wywołać funkcję, należy podać jej nazwę zakończoną nawiasem okrągłym. Jeżeli funkcja przyjmuje argumenty, należy je umieścić w tym nawiasie. W miejscu wywołania funkcji podstawiana jest zwrócona przez nią wartość. Przykładowa instrukcja:

```
$wynik = func();
```

spowoduje wywołanie funkcji o nazwie `func` i przypisanie wyniku jej działania zmiennej `$wynik`. Oto przykład wywołania funkcji z dwoma argumentami:

```
$wynik = dodaj(12, 24);
```

## Sposoby przekazywania argumentów

Funkcja może mieć dowolną liczbę argumentów oddzielonych od siebie znakiem przecinka. Argumenty przekazywane są na dwa sposoby:

- przez wartość (z ang. *by value*),
- przez referencję (z ang. *by reference*),

Argumenty mogą mieć też wartości domyślne. Standardowo przekazywanie odbywa się przez wartość, podobnie jak w innych językach programowania.

## Przekazywanie przez wartość

Przekazywanie przez wartość oznacza, że funkcja otrzymuje kopie argumentów źródłowych i wszelkie operacje wykonuje na tych kopiach. Nie jest zatem w stanie dokonać żadnej modyfikacji oryginału. Jeżeli więc jako argument została przekazana zmienna, to jej stan nie zmieni się, niezależnie od tego, co zostanie wykonane wewnątrz funkcji:

```
<?php
function f($arg){
    //modyfikacja jedynie kopii danych
    //nie ma wpływu na wartość zmiennej $zmienna
    $arg = 'dwa';
}
$zmienna = 'jeden';
echo $zmienna, "\n";
f($zmienna);
//zmienna ma wartość 'jeden'
echo $zmienna;
?>
```

## Przekazywanie przez referencję

Przekazywanie przez referencję oznacza przekazanie do funkcji oryginalnych danych. Jeżeli funkcja zmieni ich wartość (wartości), pozostanie ona zmieniona również po zakończeniu działania funkcji. Aby skorzystać z tego sposobu przekazywania argumentów, dany argument należy poprzedzić znakiem & (ampersand), schematycznie:

```
function funkcja(&argument);
```

np.:

```
<?php
function f(&$arg){
    //tutaj następuje modyfikacja
    //wartości zmiennej $zmienna
    $arg = 'dwa';
}
$zmienna = 'jeden';
echo $zmienna, "\n";
f($zmienna);
//zmienna ma wartość 'dwa'
echo $zmienna;
?>
```

## Domyślne wartości argumentów

Argumenty domyślne są definiowane podobnie jak w innych językach programowania. Należy skorzystać z konstrukcji o schematycznej postaci:



```
function nazwa_funkcji ($argument1 = wartość, $argument2 =
↳wartość2, ..., $argumentN = wartośćN)
{
    //treść funkcji
}
```

Wartość domyślna musi być wyrażeniem stałym (o stałej wartości, z ang. *constant expression*), nie może więc to być np. zmienna. Należy również pamiętać, że konieczne jest podanie wszystkich domyślnych argumentów z prawej strony listy argumentów przed wszystkimi argumentami zwykłymi, np.:

```
function show($arg1, $arg2 = "abc", $arg3 = "def")
{
    echo $arg1. " " ".$arg2." " ".$arg3;
}
```

Po takiej deklaracji funkcja `show` mogłaby być wywołana na jeden z trzech poniższych sposobów:

```
show("123");
show("123", "456");
show("123", "456", "789");
```

## Zmienna liczba argumentów

W funkcjach można stosować zmienną liczbę argumentów. Nie ma do tego żadnych specjalnych konstrukcji. Funkcja może być wywołana z dowolną liczbą argumentów, większą niż liczba podana w deklaracji. Do obsługi zmiennej listy argumentów wykorzystywane są trzy funkcje:

- `func_num_args` — zwraca liczbę argumentów funkcji,
- `func_get_arg` — zwraca argument o podanym numerze,
- `func_get_args` — zwraca listę argumentów w postaci tablicy.

Oto przykład skryptu zawierającego funkcję o zadeklarowanej zerowej liczbie argumentów, która dokonuje łączenia łańcuchów znakowych przekazanych jej w postaci argumentów:

```
<?php
function połącz(){
    $val = "";
    $count = func_num_args();
    for($i = 0; $i < $count; $i++){
        $val .= func_get_arg($i);
    }
    return $val;
}
$str = połącz("To ", "jest ", "test");
echo $str;
?>
```

A to ten sam skrypt wykorzystujący funkcję `func_get_args` i pętlę `foreach`:

```
<?php
function połącz(){
    $val = "";
    $tab = func_get_args();
    foreach($tab as $v){
        $val .= $v;
    }
    return $val;
}
$str = połącz("To ", "jest ", "test");
echo $str;
?>
```

## Zasięg zmiennych

W PHP zasięg zmiennej jest ograniczony do kontekstu, w którym została zdefiniowana. W tym znaczeniu zmienne można podzielić na:

- globalne,
- lokalne.

## Zmienne globalne

Zmienna zadeklarowana w skrypcie poza ciałem funkcji lub klasy jest zmienną globalną, tzn. dostępną bezpośrednio w każdym miejscu skryptu poza wnętrzami funkcji. W przykładowym kodzie:

```
<?php
$liczba = 100;
function f()
{
    echo $liczba;
}
f();
?>
```

zmienna `$liczba` jest globalna i nie ma do niej dostępu w funkcji `f`. Dlatego też po wywołaniu funkcji wartość nie zostanie wyświetlona, wygenerowane zostanie natomiast ostrzeżenie (poziom notice) o niezdefiniowanej zmiennej `$liczba`.

Aby w funkcji uzyskać dostęp do zmiennych o zasięgu globalnym, należy użyć słowa kluczowego `global` lub tablicy `$GLOBALS`. W pierwszym przypadku przed odwołaniem (najlepiej na początku kodu funkcji) należy użyć konstrukcji o schematycznej postaci:

```
global $zmienna1, $zmienna2,...,$zmiennaN;
```

Po wykonaniu takiej instrukcji będzie można się odwoływać do wszystkich zmiennych globalnych wymienionych po `global`, np.:

```
<?php
$liczba = 100;
function f()
{
    global $liczba;
    echo $liczba;
}
f();
?>
```

W drugim przypadku, ponieważ tablica `$GLOBALS` zawiera odwołania do wszystkich zmiennych globalnych skryptu, należy skorzystać z odwołania typu:

```
$GLOBALS['nazwa_zmiennej']
```

np.:

```
<?php
$liczba = 100;
function f()
{
    echo $GLOBALS['liczba'];
}
f();
?>
```

## Zmienne lokalne

Zasięg zmiennych lokalnych jest ograniczony wyłącznie do wnętrza funkcji, w której zostały zdefiniowane. Odwołania w innym miejscu skryptu nie są możliwe, np.:

```
<?php
function f()
{
    $liczba = 100;
    echo "Wewnątrz funkcji f: $liczba\n";
}
f();
echo "Poza funkcją f: $liczba";
?>
```

## Zmienne statyczne

Zmienne statyczne to zmienne lokalne funkcji, które zachowują swoją wartość pomiędzy jej wywołaniami. Aby zadeklarować taką zmienną, należy użyć słowa `static`, schematycznie:

```
static $nazwa_zmiennej = wartość;
```

Po takiej deklaracji (we wnętrzu funkcji) pierwsze wywołanie funkcji spowoduje utworzenie zmiennej statycznej i zapamiętanie jej ostatniej wartości, natomiast w każdym kolejnym wywołaniu instrukcja przypisująca pierwotną wartość zmiennej będzie ignorowana, a użyta zostanie wartość z poprzedniego wywołania, np.:

```
<?php
function f($val)
{
    static $liczba = 100;
    $liczba += $val;
    echo "Wartość zmiennej : $liczba\n";
}
f(10);
f(20);
?>
```

## Wartość zmiennej jako nazwa funkcji

W PHP wartość zmiennej może być potraktowana jako nazwa funkcji do wywołania. Aby skorzystać z takiej techniki, należy za nazwą zmiennej umieścić nawias okrągły, schematycznie:

```
$nazwa_zmiennej();
```

Jeżeli zatem zmiennej *\$nazwa* zostanie przypisany ciąg znaków zawierający nazwę istniejącej funkcji `wyświetl`, to po użyciu instrukcji:

```
$nazwa();
```

funkcja `wyświetl` zostanie wywołana:

```
<?php
function wyświetl()
{
    echo "To jest funkcja wyświetl.";
}

$nazwa = 'wyświetl';

$nazwa();

?>
```

W ten sam sposób mogą być wywoływane metody obiektów. Jeżeli funkcja lub metoda wymaga podania argumentów, należy je podać w nawiasie okrągłym występującym za nazwą zmiennej, np.:

```
<?php
function dodaj($arg1, $arg2)
{
    return $arg1 + $arg2;
}
```

```
$nazwa = 'dodaj';  
  
$wynik = $nazwa(2, 3);  
echo $wynik;  
?>
```

## Definiowanie funkcji wewnątrz funkcji

W PHP można definiować funkcje wewnątrz innych funkcji. Powstają wtedy funkcje wewnętrzne dostępne tylko w obrębie (w zasięgu) funkcji zewnętrznej (głównej). Nie ma limitu poziomu zagnieżdżenia funkcji wewnętrznych (tzn. jedna funkcja wewnętrzna może zawierać kolejną funkcję wewnętrzną). Schematycznie konstrukcja funkcji wewnętrznej wygląda następująco:

```
function nazwa_funkcji_zewnetrznej(argumenty){  
    function nazwa_funkcji_wewnetrznej(argumenty){  
        //treść funkcji wewnętrznej  
    }  
    //dalsza treść funkcji zewnętrznej  
}
```

Oto przykład użycia funkcji wewnętrznych:

```
<?php  
function dzialanie($val1, $val2, $op)  
{  
    function dodaj($arg1, $arg2){  
        return $arg1 + $arg2;  
    }  
    function odejmij($arg1, $arg2){  
        return $arg1 - $arg2;  
    }  
    switch($op){  
        case '+': return dodaj($val1, $val2);  
        case '-': return odejmij($val1, $val2);  
        default: return null;  
    }  
}  
  
$wynik = dzialanie(2, 3, '+');  
echo $wynik;  
?>
```

## Funkcje anonimowe

Od PHP 5.3.0 dostępne są funkcje anonimowe, czyli takie, które nie posiadają nazwy. Najczęściej używane są podczas stosowania funkcji zwrotnych (z ang. *callback functions*). Definicja funkcji anonimowej wygląda podobnie jak definicja funkcji zwykłej, schematycznie:

```
function (argument1, argument2,..., argumentN)
{
    //treść funkcji
}
```

Oto przykład użycia funkcji anonimowej jako funkcji zwrotnej:

```
<?php
function działanie($val1, $val2, $func)
{
    return $func($val1, $val2);
}

$wynik = działanie(2, 3,
    function($arg1, $arg2){
        return $arg1 + $arg2;
    }
);

echo $wynik;
?>
```

W tym przypadku trzecim argumentem funkcji `działanie` jest funkcja anonimowa przyjmująca dwa argumenty (`$arg1` i `$arg2`) i zwracająca wynik ich dodawania. Funkcja anonimowa jest wywoływana w funkcji `działanie` przez zastosowanie składni opisanej w podrozdziale „Wartość zmiennej jako nazwa funkcji”.

Funkcja anonimowa może być również przypisana bezpośrednio zmiennej, np.:

```
<?php
$zmienna = function($arg1){
    return $arg1 * 2;
};
$wynik = $zmienna(12);
echo $wynik;
?>
```

- "", 82
  - "array", 32
  - "bool", 32
  - "boolean", 32
  - "double", 32
  - "float", 32
  - "int", 32
  - "integer", 32
  - "null", 32
  - "object", 32
  - "string", 32
  - \$, 82
  - \$\_COOKIE, 28, 29
  - \$\_ENV, 29
  - \$\_FILES, 29
  - \$\_GET, 28, 29
  - \$\_POST, 28, 29, 126
  - \$\_REQUEST, 29
  - \$\_SERVER, 28
  - \$\_SESSION, 29
  - \$argc, 30
  - \$argv, 30
  - \$GLOBALS, 28
  - \$HTTP\_RAW\_POST\_DATA, 29
  - \$http\_response\_header, 29
  - \$php\_errormsg, 29
  - (array), 31
  - (bool), 30
  - (boolean), 30
  - (double), 30
  - (float), 30
  - (int), 30
  - (integer), 30
  - (object), 31
  - (real), 30
  - (string), 31
  - (unset), 31
  - (X)HTML, 17, 18
  - \, 82
  - \_\_CLASS\_\_, 40
  - \_\_DIR\_\_, 40
  - \_\_FILE\_\_, 40
  - \_\_FUNCTION\_\_, 40
  - \_\_LINE\_\_, 40
  - \_\_METHOD\_\_, 40
  - \_\_NAMESPACE\_\_, 40
  - <div>, 18
  - <script>, 16
- ## A
- addslashes, 83
  - addslashes, 84
  - apostrof, 21, 23
  - array array\_slice, 112
  - array array\_splice, 112
  - array str\_getcsv, 95
  - array\_count\_values, 109
  - array\_diff, 110
  - array\_diff\_key, 109
  - array\_fill, 110
  - array\_key\_exists, 110
  - array\_keys, 110
  - array\_pop, 111
  - array\_push, 111
  - array\_replace, 111
  - array\_reverse, 111
  - array\_search, 111
  - array\_shift, 111
  - array\_sum, 112
  - array\_unique, 112
  - array\_unshift, 113
  - arsort, 113
  - asort, 113

## B

BIG5, 87  
BIG5-HKSCS, 87  
bin2hex, 84  
bitowa różnica symetryczna, 43  
boolean, 30

## C

charset, 87  
checkdate, 116  
chop, 84  
chr, 84  
chunk\_split, 84  
closedir, 148  
Content-Disposition, 126, 127  
Content-Length, 126  
Content-Type, 126  
convert\_cyr\_string, 84  
convert\_uuencode, 85  
cookies, 129  
    obsługa, 129  
    odczyt, 131  
    usuwanie, 131  
    zapis, 129  
count\_chars, 85  
cp1251, 87  
cp1252, 87  
cp866, 87  
crc32, 85  
crypt, 86  
cudzysłów, 21, 23

## D

date, 116  
delimiter, 95  
destruktor, 138  
    tworzenie, 139  
disk\_free\_space, 154  
disk\_total\_space, 154  
dostęp  
    chroniony, 137  
    private, 137  
    protected, 137  
    prywatny, 137

    public, 137  
    publiczny, 137  
double, 20  
doubleval, 31  
dsttime, 119  
dziedziczenie, 141

## E

enclosure, 95  
ENT\_COMPAT, 86  
ENT\_NOQUOTES, 86  
ENT\_QUOTES, 86  
escape, 95  
EUCJP, 87  
EUC-JP, 87  
explode, 86  
extension\_dir, 13

## F

false, 20, 25  
feof, 155  
fgetc, 152  
fgets, 152  
fgetss, 152  
file, 153  
file\_exists, 155  
file\_get\_contents, 152  
FILE\_IGNORE\_NEW\_LINES, 153  
file\_put\_contents, 154  
FILE\_SKIP\_EMPTY\_LINES, 153  
FILE\_USE\_INCLUDE\_PATH, 153  
fileatime, 155  
filectime, 155  
filemtime, 155  
filesize, 155  
float, 20  
floatval, 31  
fopen, 150  
fpassthru, 153  
fprintf, 86  
fprintf, 86  
fputs, 154  
fscanf, 153  
fseek, 156  
ftell, 156



func\_get\_arg, 74  
func\_get\_args, 74  
func\_num\_args, 74  
function, 71  
funkcja  
  addslashes, 83  
  addslashes, 84  
  array array\_slice, 112  
  array array\_splice, 112  
  array str\_getcsv, 95  
  array\_count\_values, 109  
  array\_diff, 110  
  array\_diff\_key, 109  
  array\_fill, 110  
  array\_key\_exists, 110  
  array\_keys, 110  
  array\_pop, 111  
  array\_push, 111  
  array\_replace, 111  
  array\_reverse, 111  
  array\_search, 111  
  array\_shift, 111  
  array\_sum, 112  
  array\_unique, 112  
  array\_unshift, 113  
  arsort, 113  
  asort, 113  
  bin2hex, 84  
  checkdate, 116  
  chop, 84  
  chr, 84  
  chunk\_split, 84  
  closedir, 148  
  convert\_cyr\_string, 84  
  convert\_uuencode, 85  
  count\_chars, 85  
  crc32, 85  
  crypt, 86  
  date, 116  
  disk\_free\_space, 154  
  disk\_total\_space, 154  
  explode, 86  
  feof, 155  
  fgetc, 152  
  fgets, 152  
  fgetss, 152  
  file, 153  
  file\_exists, 155  
  file\_get\_contents, 152  
  file\_put\_contents, 154  
  fileatime, 155  
  filectime, 155  
  filemtime, 155  
  filesize, 155  
  fopen, 150  
  fpassthru, 153  
  fprintf, 86  
  fputs, 154  
  fscanf, 153  
  fseek, 156  
  ftell, 156  
  fwrite, 154  
  getdate, 118  
  gettimeofday, 119  
  gmdate, 119  
  gmmktime, 119  
  gmstrftime, 119  
  html\_entity\_decode, 86  
  htmlentities, 87  
  htmlspecialchars, 88  
  htmlspecialchars\_decode, 87  
  idate, 119  
  implode, 88  
  in\_array, 113  
  join, 88  
  ksort, 113  
  ksort, 114  
  lcfirsr, 88  
  levenshtein, 88  
  localeconv, 89  
  localtime, 120  
  ltrim, 89  
  md5, 89  
  md5\_file, 89  
  metaphone, 89  
  microtime, 120  
  mktime, 121  
  money\_format, 90  
  natcasesort, 114  
  natsort, 114  
  nl2br, 90  
  number\_format, 90  
  opendir, 148  
  ord, 91

## funkcja

- parse\_str, 91
- printf, 91
- quoted\_printable\_decode, 91
- quoted\_printable\_encode, 91
- quotemeta, 91
- range, 114
- readdir, 148
- readfile, 153
- rsort, 114
- rtrim, 91
- setlocale, 92
- sha1, 93
- sha1\_file, 92
- shuffle, 115
- similar\_text, 93
- sort, 115
- soundex, 93
- sprintf, 93
- sscanf, 95
- str\_ireplace, 95
- str\_pad, 96
- str\_repeat, 96
- str\_replace, 96
- str\_rot13, 96
- str\_shuffle, 96
- str\_split, 96
- str\_word\_count, 97
- strcasecmp, 97
- strchr, 97
- strcmp, 97
- strcoll, 97
- strcspn, 98
- strftime, 121
- string convert\_uencode, 85
- strip\_tags, 98
- stripclashes, 98
- stripos, 98
- stripslashes, 98
- stristr, 98
- strlen, 98
- strnatcasecmp, 99
- strnatcmp, 99
- strncasecmp, 99
- strncmp, 99
- strpbrk, 99
- strpos, 99

- strptime, 123
- strrchr, 99
- strrev, 100
- stripos, 100
- strpos, 100
- strspn, 100
- strstr, 100
- strtok, 101
- strtolower, 101
- strtotime, 124
- strtoupper, 101
- strtr, 101
- substr, 102
- substr\_compare, 101
- substr\_count, 102
- substr\_replace, 102
- time, 124
- trim, 102
- uasort, 115
- ucfirst, 102
- ucwords, 103
- uksort, 115
- usort, 115
- vfprintf, 103
- vprintf, 103
- vsprintf, 103
- wordwrap, 103

## funkcje, 71

- anonimowe, 78
- deklaracja, 71
- domyślne wartości argumentów, 73
- operująca na systemie plików, 154
- przekazywanie argumentów
  - przez referencję, 73
  - przekazywanie argumentów
    - przez wartość, 73
  - przekazywanie argumentów, 72
  - wracanie wartości, 71
  - wywołanie, 72
  - zmienna liczba argumentów, 74
- fwrite, 154

## G

- GB2312, 87
- getdate, 118
- gettimeofday, 119

gettype(), 35  
gmdate, 119  
gmmktime, 119  
gmstrftime, 119

## H

heredoc, 21  
hours, 118  
HTML 4.01 Strict, 17  
html\_entity\_decode, 86  
htmlentities, 87  
htmlspecialchars, 88  
htmlspecialchars\_decode, 87  
httpd.conf, 11, 12

## I

ibm866, 87  
idate, 119  
identyfikatory, 25  
if, 20  
iloczyn bitowy, 42  
implode, 88  
in\_array, 113  
include, 69, 70  
include\_once, 69, 70  
include\_path, 13, 70  
index.php, 11, 14  
instalacja  
    Linux, 9  
        integracja z serwerem  
            Apache, 11  
        za pomocą gotowych  
            pakietów, 9  
        ze źródeł, 10  
    Windows, 11  
        instalator, 12  
        integracja z serwerem  
            Apache, 12  
        ręczna, 12  
instrukcja  
    break, 62  
    continue, 64  
    goto, 65  
    if...else if, 57, 67

if...else, 57, 66  
wyboru switch, 58  
sterująca, 57  
warunkowa, 57  
int, 20  
integer, 20  
intval, 31  
is\_array(), 35  
is\_bool(), 35  
is\_double(), 35  
is\_float(), 35  
is\_int(), 35  
is\_integer(), 35  
is\_long(), 35  
is\_null(), 35  
is\_numeric, 35  
is\_object(), 35  
is\_real(), 35  
is\_resource, 35  
is\_scalar, 35  
is\_string(), 35  
ISO-8859-1, 87  
ISO-8859-15, 87  
ISO-8859-2, 17

## J

join, 88

## K

katalog  
    odczytywanie, 148  
    otwieranie, 148  
    tworzenie, 149  
    usuwanie, 149  
    zamykanie, 148  
klasa  
    definiowanie, 136  
    Exception, 145  
    finalna, 143  
    wyjątków, 147  
klonowanie obiektów, 53  
kod ASCII  
    0x00, 89  
    0x09, 89  
    0x0A, 89

- kod ASCII
  - 0x0B, 89
  - 0x0D, 89
  - 0x32, 89
- koi8r, 87
- koi8-ru, 87
- komentarz
  - blokowy, 18
  - jednowierszowy, 18
    - uniksowy, 18, 19
    - zwykły, 19
- konstruktory, 138
  - argumenty, 139
  - tworzenie, 138
- kontrola typów danych, 34
- konwersja typów, 30
  - automatyczna, 30
    - do typu całkowitego (integer), 33
    - do typu logicznego (boolean), 32
    - do typu łańcuchowego (string), 34
    - do typu zmiennoprzecinkowego (double), 33
  - wymuszona, 30
  - zasady, 32
- ksort, 113
- ksort, 114

## L

- LC\_ALL, 92
- LC\_COLLATE, 92
- LC\_CTYPE, 92
- LC\_MESSAGES, 92
- LC\_MONETARY, 92
- LC\_NUMERIC, 92
- LC\_TIME, 92
- lcfirst, 88
- levenshtein, 88
- lewy ukośnik, 82
- liczby zmiennopozycyjne, 20
- liczby zmiennoprzecinkowe, 20
- literały, 22
  - null, 22, 25
  - całkowite, 22
  - logiczne, 22, 25
  - łańcuchowe, 22, 23

- rzeczywiste, 22
- zmiennopozycyjne, 22
- zmiennoprzecinkowe, 22
- localeconv, 89
- localtime, 120
- ltrim, 89

## M

- max\_execution\_time, 13
- md5, 89
- md5\_file, 89
- mday, 118
- memory\_limit, 13
- metaphone, 89
- metoda, 136
  - finalna, 142
  - GET, 125
  - getCode, 145
  - getFile, 145
  - getLine, 145
  - getMessage, 145
  - getPrevious, 145
  - getTrace, 146
  - getTraceAsString, 146
  - POST, 126
  - przeciążanie, 140
- microtime, 120
- minutes, 118
- minuteswest, 119
- mktime, 121
- modyfikatory dostępu, 137
- mon, 118
- money\_format, 90
- month, 118

## N

- n, 82
- natcasesort, 114
- natsort, 114
- negacja bitowa, 42, 43
- nl2br, 90
- nnn, 82
- nowa linia, 82
- nowdoc, 21
- number\_format, 90

## O

obiekty  
klonowanie, 53  
tworzenie, 52, 137  
obsługa standardowego wyjścia, 80  
odwołania do stałych, 37  
określenie wersji PHP, 41  
opendir, 148  
operatory, 40  
  arytmetyczne, 40, 42  
  dodawanie, 42  
  dzielenie, 42  
  dzielenie modulo, 42  
  mnożenie, 42  
  odejmowanie, 42  
  reszta z dzielenia, 42  
bitowe, 40, 42  
  alternatywa wykluczająca, 42  
  bitowa różnica symetryczna, 42  
  iloczyn, 42  
  negacja bitowa, 42  
  operacja AND, 42  
  operacja NOT, 42  
  operacja OR, 42  
  operacja XOR, 42  
  przesunięcie bitowe w lewo, 42  
  przesunięcie bitowe w prawo, 42  
  suma bitowa, 42  
dekrementacji, 40, 48  
indeksowania tablic, 49  
inkrementacji, 40, 48  
kontroli błędów, 51  
kontroli typów, 51  
logiczne, 40, 45  
  alternatywa logiczna, 46  
  iloczyn logiczny, 45  
  logiczna alternatywa  
  wykluczająca, 46  
  negacja logiczna, 46  
  różnica symetryczna, 46  
  suma logiczna, 46  
łańcuchowe, 40, 50  
łączenia tablic, 49  
porównywania, 40, 47, 49  
pozostałe, 40

  priorytety, 53  
  przypisania, 40, 47  
  relacyjne, 40, 47  
  rzutowania typów, 52  
  tablicowe, 48  
  warunkowe, 40, 50  
ord, 91

## P

parse\_str, 91  
pętla, 59  
  do...while, 59, 60  
  for, 59, 67  
  foreach, 59, 61, 68  
  instrukcja break, 62  
  instrukcja continue, 64  
  instrukcja goto, 65  
  instrukcja switch, 68  
  while, 59, 60, 68  
  zagnieżdżanie, 62  
PHP, 7  
  Personal HomePage Toolkit, 7  
  Personal HomePage Tools, 7  
  PHP Hypertext Preprocessor, 7  
php.ini, 10, 12, 13, 16, 126, 127, 133  
PHP5. Praktyczny kurs. Wydanie II, 8  
plik  
  odczyt, 152  
  otwieranie, 150  
  tworzenie, 150  
  zapis danych, 153  
  usuwanie, 150  
  zamykanie, 150  
plik konfiguracyjny, 13 *Patrz też*  
  php.ini  
pola, 136  
polecenie zewnętrzne, 52  
post\_max\_size, 13  
powrót karetki, 82  
printf, 91, 94  
private, 137  
protected, 137  
przeciążanie, 140  
  metod, 140  
przesłanianie składowych, 142

- przesunięcie
  - bitowe w lewo, 42, 44
  - bitowe w prawo, 42, 44
- strony, 82
- wysunięcie, 82
- przypisanie wartości do zmiennej, 26
- public, 137

## Q

- quot\_style, 86
- quoted\_printable\_decode, 91
- quoted\_printable\_encode, 91
- quotemeta, 91

## R

- range, 114
- readdir, 148
- readfile, 153
- require, 69, 70
- require\_once, 69, 70
- return, 71
- różnica bitowa, 42
- rsort, 114
- rtrim, 91
- rzutowanie, 52

## S

- sec, 119
- seconds, 118
- sekcja try...catch, 146
- sesja
  - identyfikator, 131
  - kończenie, 132
  - obsługa, 131
  - rozpoczynanie, 132
  - zmiennie, 132
- session.entropy\_length, 134
- session.entropy\_file, 134
- session.serialize\_handler, 135
- session.auto\_start, 133
- session.cache\_expire, 133
- session.cookie\_domain, 133
- session.cookie\_httponly, 133

- session.cookie\_lifetime, 134
- session.cookie\_path, 134
- session.cookie\_secure, 134
- session.gc\_divisor, 134
- session.gc\_maxlifetime, 134
- session.gc\_probability, 134
- session.hash\_bits\_per\_character, 134
- session.hash\_function, 134
- session.name, 134
- session.referer\_check, 135
- session.save\_handler, 135
- session.save\_path, 135
- session.use\_cookies, 135
- session.use\_only\_cookies, 135
- session.use\_trans\_sid, 135
- setcookie, 129
- setlocale, 92
- settype, 31
- sha1, 93
- sha1\_file, 92
- Shift\_JIS, 87
- shuffle, 115
- similar\_text, 93
- SJIS, 87
- składnia
  - heredoc, 21, 23, 24, 81
  - nowdoc, 21, 23, 24, 81
- składowe
  - klasy, 136
  - statyczne, 143
- skrypt, 15, 17
  - łączenie, 69
- słowa kluczowe, 25
  - abstract, 26
  - and, 26
  - array, 26
  - as, 26
  - break, 26
  - case, 26
  - catch, 26
  - cfunction, 26
  - class, 26
  - clone, 26
  - const, 26
  - continue, 26
  - declare, 26
  - default, 26

- do, 26
- else, 26
- elseif, 26
- enddeclare, 26
- endfor, 26
- endforeach, 26
- endif, 26
- endswitch, 26
- endwhile, 26
- extends, 26
- final, 26
- for, 26
- foreach, 26
- function, 26
- global, 26
- goto, 26
- if, 26
- implements, 26
- instanceof, 26
- interface, 26
- namespace, 26
- new, 26
- old\_function, 26
- or, 26
- private, 26
- protected, 26
- public, 26
- static, 26
- switch, 26
- throw, 26
- try, 26
- use, 26
- var, 26
- while, 26
- xor, 26
- sort, 115
  - `SORT_LOCALE_STRING`, 112
  - `SORT_NUMERIC`, 112
  - `SORT_REGULAR`, 112
  - `SORT_STRING`, 112
- soundex, 93
- specyfikatory dostępu, 137
- sprintf, 93, 94
- scanf, 95
- stała, 36
  - definiowanie, 36
  - magiczna, 40
  - napisowa, 22
- odwołania, 37
- predefiniowana, 40, 41
  - `DEFAULT_INCLUDE_PATH`, 41
  - `PHP_CONFIG_FILE_PATH`, 41
  - `PHP_EOL`, 41
  - `PHP_EXTENSION_DIR`, 41
  - `PHP_INT_MAX`, 41
  - `PHP_INT_SIZE`, 41
  - `PHP_MAJOR_VERSION`, 41
  - `PHP_MINOR_VERSION`, 41
  - `PHP_OS`, 41
  - `PHP_RELEASE_VERSION`, 41
  - `PHP_VERSION`, 41
  - `PHP_VERSION_ID`, 41
  - `PHP_WINDOWS_VERSION_BUILD`, 41
  - `PHP_WINDOWS_VERSION_MAJOR`, 41
  - `PHP_WINDOWS_VERSION_MINOR`, 41
- standardowe wyjście, 80
- `str_ireplace`, 95
- `str_pad`, 96
  - `STR_PAD_BOTH`, 96
  - `STR_PAD_LEFT`, 96
  - `STR_PAD_RIGHT`, 96
- `str_repeat`, 96
- `str_replace`, 96
- `str_rot13`, 96
- `str_shuffle`, 96
- `str_split`, 96
- `str_word_count`, 97
- `strcasecmp`, 97
- `strchr`, 97
- `strcmp`, 97
- `strcoll`, 97
- `strcspn`, 98
- `strftime`, 121
- `string convert_uuencode`, 85
- `strip_tags`, 98
- `stripslashes`, 98
- `stripos`, 98
- `stripslashes`, 98
- `stristr`, 98
- `strlen`, 98
- `strnatcasecmp`, 99
- `strnatcmp`, 99

strncasecmp, 99  
strncmp, 99  
strpbrk, 99  
strpos, 99  
strptime, 123  
strchr, 99  
strrev, 100  
strrips, 100  
strrpos, 100  
strspn, 100  
strstr, 100  
strtok, 101  
strtolower, 101  
strtotime, 124  
strtoupper, 101  
strtr, 101  
strval, 31  
substr, 102  
substr\_compare, 101  
substr\_count, 102  
substr\_replace, 102  
suma bitowa, 42, 43

## T

tablice, 104  
  asocjacyjne, 105  
  indeksowane numerycznie, 104  
  rozmiar, 107  
  wielowymiarowe, 106  
tabulator, 82  
  pionowy, 82  
  poziomy, 82  
time, 124  
tm\_hour, 120, 123  
tm\_isdst, 120  
tm\_mday, 120, 123  
tm\_min, 120, 123  
tm\_mon, 120, 123  
tm\_sec, 120, 123  
tm\_wday, 120, 123  
tm\_yday, 120, 123  
tm\_year, 120, 123  
trim, 102  
true, 20, 25  
try...catch, 146  
tworzenie obiektów, 52

tworzenie wyjątków, 147  
typ danych, 19  
  boolean, 19, 20  
  boolowski, 32  
  całkowitoliczbowy, 20, 30, 32  
  double, 19, 20, 31  
  float, 19, 20  
  integer, 19, 20, 31  
  kontrola, 34  
  łańcuchowy, 20, 31, 32  
  NULL, 21, 31, 32  
  obiektowy, 21, 31, 32  
  prosty, 19  
  resource, 21  
  skalarny, 19  
  specjalny, 19, 21  
  string, 19, 20, 31  
  tablicowy, 31, 32  
  złożony, 19, 21  
  zmiennoprzecinkowy, 30, 32

## U

uasort, 115  
ucfirst, 102  
ucwords, 103  
uksort, 115  
unparsed, 123  
upload\_max\_filesize, 13  
usec, 119  
usort, 115  
UTF-8, 17, 87  
uuencode, 85

## V

vfprintf, 103  
vprintf, 94, 103  
vsprintf, 103

## W

wday, 118  
weekday, 118  
wiersz poleceń, 14



Windows-1251, 87  
Windows-1252, 87  
właściwości, 136  
wordwrap, 103  
wyjątki predefiniowane, 147  
wyrażenia, 54

## X

XHTML 1.0, 17  
xNN, 82

## Y

yday, 118  
year, 118

## Z

zgłaszanie wyjątków, 145  
zmiennie, 26  
  autoglobalne, 28  
  globalne, 28, 75  
  lokalne, 76  
  przypisanie wartości, 26  
  statyczne, 76  
  superglobalne, 28  
  typ, 26  
znaczniki, 15  
  (X)HTML, 81  
  kanoniczne, 15, 16  
  skryptów HTML, 15, 16  
  typu ASP, 15, 16  
  typu SGML, 15, 16  
znak cudzysłowu, 82  
znak dolara, 82  
znaki specjalne, 82

# PHP 5

## LEKSYKON KIESZONKOWY

Język skryptowy PHP od lat wciąż na nowo udowadnia swoją elastyczność i niezawodność. Wykorzystywany do tworzenia dynamicznych serwisów internetowych, a także samodzielnych aplikacji, szczególnie dobrze sprawdza się wtedy, gdy chcemy zapewnić możliwość współdziałania naszej witryny z różnymi formatami danych oraz skomplikowanymi bazami. Ponadto jego opanowanie nie nastęrcza wielu trudności, a efekty pracy nawet początkujących webmasterów bywają wręcz spektakularne. Jeśli masz już za sobą pierwsze kroki w dziedzinie programowania w PHP 5, w codziennej pracy z pewnością przyda Ci się poręczna ściągą, w której zawarto najważniejsze informacje i sposoby wykonywania podstawowych zadań programistycznych. Właśnie trzymasz ją w ręku!

„PHP 5. Leksykon kieszonkowy” powstał po to, by ułatwić Ci szybkie przypomnienie sobie wybranych zagadnień związanych z tą tematyką. Znajdziesz tu informacje o instalacji oraz konfiguracji środowiska w systemach Linux i Windows, strukturze języka, instrukcjach sterujących czy funkcjach. Przeczytasz o operacjach na danych, a także o obsłudze ciągu znaków daty i czasu. Przypomnisz sobie, jak używać tablic, współpracować z serwerem i przeglądarką, obsługiwać wyjątki. W leksykonie odszukasz także wiadomości dotyczące systemu plików i programowania obiektowego. Wszystko to wydatnie skróci Twoją pracę i pomoże Ci uniknąć wielu potencjalnych błędów.

**Dobra podpowiedź zawsze pod ręką!**

- Instalacja i konfiguracja
- Struktura języka
- Instrukcje sterujące
- Funkcje
- Operacje na danych i obsługa ciągu znaków
- Tablice
- Obsługa daty i czasu
- Współpraca z serwerem i przeglądarką
- Programowanie obiektowe
- Wyjątki
- Obsługa systemu plików

Nr katalogowy: 6793



Księgarnia internetowa:  
<http://helion.pl>



Zamówienia telefoniczne:  
**0 801 339900**



**0 601 339900**



**Helion**

Sprawdź najnowsze promocje:

● <http://helion.pl/promocje>

Książki najchętniej czytane:

● <http://helion.pl/bestsellery>

Zamów informacje o nowościach:

● <http://helion.pl/news>

**Helion SA**

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: [helion@helion.pl](mailto:helion@helion.pl)

<http://helion.pl>

**helion.pl**  
księgarnia  
internetowa

**Cena 24,90 zł**

ISBN 978-83-246-3394-4



9 788324 633944

**Informatyka w najlepszym wydaniu**