

```
<?php
```

```
    $objFoo = new Foo;
```

```
<input type="submit"
```

```
<?php echo ($_SERVER['SCRIPT_NAME'])
```

```
) else {
```

```
    echo
```

```
    try
```

```
    $pdo
```

# PHP5

## Praktyczny kurs

**Rewelacyjne strony internetowe w PHP 5** — dowiedz się, jak je zrobić!

- Instalacja i konfiguracja narzędzi, czyli bez czego nie da się ruszyć dalej
- Elementy języka i współpraca z systemem plików, czyli co koniecznie trzeba opanować
- Obiektowy PHP i współpraca z bazami danych, czyli co może Ci się przydać

**Wydanie II**

Marcin Lis



Zawiera CD



## » 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

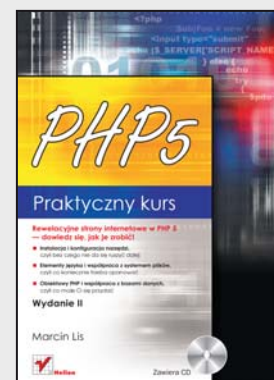
- Fragmenty książek online

## » Kontakt

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

## PHP 5. Praktyczny kurs. Wydanie II

Autor: [Marcin Lis](#)  
ISBN: 978-83-246-3393-7  
Format: 158×235, stron: 456



### Rewelacyjne strony internetowe w PHP 5 – dowiedz się, jak je zrobić!

- Instalacja i konfiguracja narzędzi, czyli bez czego nie da się ruszyć dalej
- Elementy języka i współpraca z systemem plików, czyli co koniecznie trzeba opanować
- Obiektowy PHP i współpraca z bazami danych, czyli co może Ci się przydać

Znajomość języka skryptowego PHP, szczególnie w wersji PHP 5, to w dzisiejszych czasach standard, bez którego nie może się obyć żaden szanujący się twórca stron internetowych. Środowisko PHP 5 pozwala tworzyć dynamiczne witryny, efektywnie komunikujące się z bazami danych. Zapewnia też możliwość współpracy Twojej strony z różnymi rodzajami danych, a ponadto nadaje się do tworzenia samodzielnie działających aplikacji. Jeśli marzy Ci się kariera webmastera, a nie masz jeszcze odpowiednich umiejętności w tej dziedzinie lub chcesz odświeżyć wiedzę, ta książka umożliwi Ci szybkie wejście na grunt praktycznego zastosowania możliwości języka PHP w Twoich własnych projektach.

W podręczniku „PHP 5. Praktyczny kurs. Wydanie II” znajdziesz kompletne informacje o podstawach PHP – od kwestii związanych z nazewnictwem, instalacją i konfiguracją niezbędnych narzędzi, przez omówienie zasad budowy skryptów, aż po programowanie obiektowe i obsługę sieci. Dowiesz się, jak uruchomić działającą (i atrakcyjną) witrynę internetową, jak wykorzystać możliwości grafiki i o czym należy pamiętać, aby uniknąć kłopotów z wyświetlaniem witryny w przeglądarce. Nauczysz się obsługiwać protokoły sieciowe i zrozumiesz, na czym polega obsługa sesji. Sprawdź, jak wiele możesz osiągnąć, programując w PHP!

- Instalacja i konfiguracja narzędzi
- Pierwszy skrypt
- Zmienne, typy danych i operatory
- Instrukcje sterujące i funkcje
- Obsługa daty i czasu, ciągi znaków, tablice
- Operacje na strukturze systemu plików
- Operacje na plikach i praktyczne wykorzystanie plików
- Odbieranie danych z przeglądarki
- Wysyłanie danych do przeglądarki
- Obsługa cookies, sesje i wyjątki
- Programowanie obiektowe
- Obsługa grafiki
- Połączenia sieciowe, poczta i FTP
- PHP i popularne bazy danych
- Obiektowa współpraca z MySQL i SQLite
- Bazy danych w praktyce

**PHP 5 – po prostu musisz to znać!**

# Spis treści

<b>Wstęp</b> .....	<b>7</b>
<b>Rozdział 1. Podstawy</b> .....	<b>11</b>
Lekcja 1. Czym jest PHP? .....	11
Język skryptowy .....	11
Krótka historia PHP .....	12
Jak to działa? .....	12
Lekcja 2. Instalacja i konfiguracja narzędzi .....	14
Samodzielna instalacja narzędzi .....	14
Korzystanie z pakietu XAMPP .....	31
Plik konfiguracyjny PHP .....	38
PHP w wierszu poleceń .....	39
<b>Rozdział 2. Elementy języka</b> .....	<b>41</b>
Lekcja 3. Pierwszy skrypt .....	41
Zaczynamy .....	41
Znaczniki PHP .....	45
Komentarze .....	47
Wyświetlanie informacji .....	49
Instrukcja print .....	50
Łączenie skryptów .....	51
Lekcja 4. Zmienne, typy danych i operatory .....	57
Czym są zmienne? .....	57
Rodzaje zmiennych, czyli typy danych .....	58
Zmienne w kodzie skryptu .....	62
Wyświetlanie wartości zmiennych .....	64
Operacje na zmiennych .....	68
Operatory .....	68
Zmienne globalne (superglobalne) .....	83
Konwersje typów .....	85
Ćwiczenia do samodzielnego wykonania .....	89
Lekcja 5. Instrukcje sterujące .....	90
Instrukcje warunkowe .....	90
Instrukcja wyboru .....	97
Operator warunkowy .....	99
Pętle .....	100
Składnia alternatywna .....	111
Ćwiczenia do samodzielnego wykonania .....	113

Lekcja 6. Funkcje .....	114
Definiowanie funkcji .....	114
Argumenty funkcji .....	115
Zwracanie wartości przez funkcje .....	117
Zasięg zmiennych .....	118
Sposoby przekazywania argumentów .....	123
Domyślne argumenty funkcji .....	125
Ćwiczenia do samodzielnego wykonania .....	126
Lekcja 7. Obsługa daty i czasu .....	127
Wyświetlanie daty i czasu .....	127
Tworzenie znacznika czasu .....	135
Pozostałe funkcje .....	138
Ćwiczenia do samodzielnego wykonania .....	141
Lekcja 8. Ciągi znaków .....	141
Rodzaje ciągów znaków .....	141
Wyrażenia złożone w ciągach znaków .....	143
Która metoda jest szybsza? .....	144
Formatowanie ciągów .....	146
Przetwarzanie ciągów znaków .....	152
Porównania .....	154
Przeszukiwanie .....	155
Ćwiczenia do samodzielnego wykonania .....	157
Lekcja 9. Tablice .....	158
Proste tablice .....	158
Tablice asocjacyjne .....	161
Operacje na tablicach .....	163
Ćwiczenia do samodzielnego wykonania .....	170
<b>Rozdział 3. Współpraca z systemem plików .....</b>	<b>171</b>
Lekcja 10. Operacje na strukturze systemu plików .....	171
Odczyt zawartości katalogu .....	171
Operacje na katalogach .....	176
Operacje na plikach .....	178
Miejsce na dysku .....	180
Rekurencyjne usuwanie zawartości katalogu .....	182
Nawigacja po katalogach .....	182
Ćwiczenia do samodzielnego wykonania .....	185
Lekcja 11. Operacje na plikach .....	186
Tworzenie i otwieranie plików .....	186
Zamykanie plików .....	188
Odczyt danych .....	188
Zapis danych .....	195
Inne operacje .....	201
Ćwiczenia do samodzielnego wykonania .....	204
Lekcja 12. Praktyczne wykorzystanie plików .....	204
Tekstowy licznik odwiedzin .....	205
Licznik wykorzystujący grafikę .....	207
Lista odnośników .....	209
Lista odwiedzin .....	210
Ćwiczenia do samodzielnego wykonania .....	213

<b>Rozdział 4. Współpraca z przeglądarką .....</b>	<b>215</b>
Lekcja 13. Odbieranie danych z przeglądarki .....	215
Formularze HTML .....	215
Wysyłanie metodą GET .....	216
Metoda POST .....	221
Wysyłanie plików do serwera .....	223
Ćwiczenia do samodzielnego wykonania .....	227
Lekcja 14. Wysyłanie danych do przeglądarki .....	228
Sposoby wysyłania danych .....	228
Wysyłanie zawartości plików .....	228
Sposoby pobierania plików z serwisa .....	232
Ćwiczenia do samodzielnego wykonania .....	245
Lekcja 15. Obsługa cookies .....	245
Krótko o cookies .....	245
Obsługa cookies w PHP .....	246
Korzystanie z cookies .....	250
Ćwiczenia do samodzielnego wykonania .....	255
Lekcja 16. Sesje .....	255
Wstęp do sesji .....	255
Identyfikator sesji .....	256
Rozpoczynanie sesji .....	256
Kończenie sesji .....	257
Zmienne sesji .....	257
Konfiguracja sesji .....	258
Implementacja sesji .....	260
Śledzenie zachowań użytkownika .....	264
Kontrola dostępu z wykorzystaniem sesji .....	266
System logowania z danymi w pliku .....	271
Ćwiczenia do samodzielnego wykonania .....	274
<b>Rozdział 5. Obiektowy PHP .....</b>	<b>275</b>
Lekcja 17. Podstawy obiektowości .....	275
Czym jest obiekt? .....	275
Definicja klasy .....	276
Tworzenie obiektów .....	279
Konstruktory i destruktory .....	281
Automatyczne ładowanie kodu klasy .....	284
Obiektowa lista odwiedzin .....	286
Ćwiczenia do samodzielnego wykonania .....	288
Lekcja 18. Więcej o programowaniu obiektowym .....	288
Dziedziczenie .....	288
Przesłanianie składowych .....	292
Klasy i składowe finalne .....	294
Konstruktory i destruktory klas bazowych .....	295
Specyfikatory dostępu .....	297
Składowe statyczne .....	299
Ćwiczenia do samodzielnego wykonania .....	302
Lekcja 19. Wyjątki .....	303
Instrukcja throw .....	303
Klasa Exception i pochodne .....	304
Blok try...catch .....	305
Przechwytywanie wielu wyjątków .....	310
Własne wyjątki .....	313
Ćwiczenia do samodzielnego wykonania .....	314

<b>Rozdział 6. Grafika i obrazy .....</b>	<b>315</b>
Lekcja 20. Obsługa grafiki .....	315
Biblioteka graficzna .....	315
Jak stworzyć galerię obrazów? .....	316
Przetwarzanie grafiki .....	325
Ćwiczenia do samodzielnego wykonania .....	337
<b>Rozdział 7. Obsługa sieci .....</b>	<b>339</b>
Lekcja 21. Połączenia, poczta i FTP .....	339
Tablica \$_SERVER .....	339
Adresy IP .....	342
Jak rozpoznać przeglądarkę? .....	345
Połączenie FTP .....	347
Wysyłanie poczty .....	349
Ćwiczenia do samodzielnego wykonania .....	352
<b>Rozdział 8. Współpraca z bazami danych .....</b>	<b>353</b>
Lekcja 22. Podstawy baz danych .....	353
MySQL i SQLite .....	353
Tabele, klucze i relacje .....	354
Bazy danych a PHP .....	358
Instalacja systemu bazy danych .....	359
Obsługa serwera MySQL .....	365
Lekcja 23. Podstawy SQL .....	371
Czym jest SQL? .....	371
Obsługa tabel .....	372
Typy danych w kolumnach .....	377
Zapytania .....	382
Lekcja 24. PHP i bazy danych .....	394
PHP i MySQL .....	394
PHP i SQLite .....	402
Ćwiczenia do samodzielnego wykonania .....	409
Lekcja 25. Podejście obiektowe .....	410
Korzystanie z PDO .....	410
PHP i SQLite .....	418
Lekcja 26. Bazy danych w praktyce .....	423
Licznik .....	423
Logowanie .....	426
Ankieta .....	428
Lista odwiedzin .....	433
Liczba osób na stronie .....	435
Ćwiczenia do samodzielnego wykonania .....	437
<b>Skorowidz .....</b>	<b>439</b>

## Rozdział 6.

# Grafika i obrazy

## Lekcja 20. Obsługa grafiki

### Biblioteka graficzna

PHP umożliwia zarówno wykonywanie różnorodnych operacji na obrazach, jak i generowanie plików graficznych. Wraz z PHP jest w tym celu dostarczana standardowo biblioteka GD<sup>1</sup> (dostępna również pod adresem <http://www.libgd.org/> lub <http://www.boutell.com/gd/>). W przypadku systemu Windows uaktywnienie funkcji graficznych uzyskamy przez usunięcie komentarza (początkowego średnika) z wiersza:

```
extension=php_gd2.dll
```

znajdującego się w pliku *php.ini* (oczywiście o ile ten wiersz jest ujęty w komentarz — poprzedzony średnikiem).

W przypadku Linuksa w pakietach dla poszczególnych dystrybucji biblioteka GD jest zwykle dołączana standardowo i nie trzeba wykonywać dodatkowych czynności. Aby włączyć obsługę biblioteki GD w przypadku samodzielnej kompilacji pakietów PHP, należy podczas konfiguracji podać dodatkową opcję:

```
--with-gd
```

jeśli chcemy korzystać z biblioteki dostarczonej wraz z PHP lub:

```
--with-gd=nazwa_katalogu
```

jeśli chcemy korzystać z zewnętrznej wersji biblioteki, np. pobranej spod podanego wyżej adresu. W tym przypadku parametr *nazwa\_katalogu* powinien wskazywać katalog, w którym ta biblioteka została zainstalowana.

---

<sup>1</sup> Jako rozszerzenia dostępne są też inne biblioteki, np. *ImageMagic*, *Gmagick*, *Cairo*. Dokładne informacje o nich można znaleźć w dokumentacji PHP.

W celu przekonania się, czy dana wersja PHP ma włączoną obsługę biblioteki GD, można użyć funkcji o nazwie `gd_info`. Zwraca ona tablicę asocjacyjną zawierającą informacje o konfiguracji. Wystarczy zatem wywołać skrypt przedstawiony na listingu 6.1.

**Listing 6.1.** *Informacje o zainstalowanej bibliotece graficznej*

```
<?php
$arr = gd_info();
foreach($arr as $key => $val){
    echo "$key = $val\n";
}
?>
```

Przykładowy efekt działania kodu w konsoli systemu Windows został zaprezentowany na rysunku 6.1 (skrypt można też wywołać w przeglądarce, dodając jednak dla formatowania znacznik `<br />`). W sytuacji gdy wykonanie skryptu nie spowoduje wyświetlenia informacji lub też pojawi się komunikat o nieznannej funkcji `gd_info`, będzie to oznaczać, że obsługa biblioteki GD nie została włączona.

**Rysunek 6.1.**  
*Informacje o wersji i opcjach biblioteki graficznej GD*

```
E:\WINDOWS\system32\cmd.exe

E:\>php index.php
GD Version = bundled (2.0.34 compatible)
FreeType Support = 1
FreeType Linkage = with freetype
TLib Support =
GIF Read Support = 1
GIF Create Support = 1
JPEG Support = 1
PNG Support = 1
WBMP Support = 1
XPM Support =
XBM Support = 1
JIS-mapped Japanese Font Support =

E:\>_
```

## Jak stworzyć galerię obrazów?

Zanim przejdziemy do procedur przetwarzających grafikę, spróbujmy zająć się bardzo popularnym tematem tworzenia galerii obrazów. W większości wypadków nie wymaga to wykorzystywania funkcji udostępnianych przez bibliotekę GD — wystarczą zdobyte do tej pory wiadomości dotyczące języka PHP, w tym współpracy z przeglądarką (lekcje z rozdziału 4.) oraz systemem plików (lekcje z rozdziału 3.). Stwórzmy więc kilka przykładowych galerii.

### Prosta galeria

Najprostsza galeria wyświetla pojedyncze obrazy wraz z odnośnikami nawigacyjnymi, tak jak zostało to zaprezentowane na rysunku 6.2. Jej utworzenie nie powinno sprawić żadnego kłopotu. Pliki z obrazami zapiszemy w osobnym katalogu, np. o nazwie *images*. Skrypt oczywiście uwzględni automatycznie wszelkie zmiany zawartości tego katalogu, nie trzeba więc będzie troszczyć się o ręczną aktualizację danych. Aby jednak nie komplikować kodu skryptu, przyjęte zostanie założenie, że katalog nie może





**Rysunek 6.2.** Przykładowy wygląd galerii

zawierać innych plików niż pliki galerii ani podkatalogów. Będzie się w nim też musiał znajdować co najmniej jeden plik graficzny.

Numer wyświetlanego obrazu będzie przekazywany do skryptu za pomocą metody GET w postaci parametru o nazwie `imgid`. Obrazy będą wyświetlane w kolejności alfabetycznej (według nazw plików), pierwszy z nich będzie miał identyfikator 0, drugi — 1 itd. Na każdej stronie galerii, tak jak jest to widoczne na rysunku 6.2, będą wyświetlane:

- ♦ obraz (umieszczany na stronie za pomocą znacznika `<img>`),
- ♦ nazwa pliku graficznego,
- ♦ numer obrazu (numeracja będzie się rozpoczynać od 1),
- ♦ całkowita liczba obrazów,
- ♦ odnośniki do pierwszego, poprzedniego, następnego i ostatniego obrazu.

Skrypt generujący tak określoną galerię został przedstawiony na listingu 6.2. Przyjrzyjmy się mu bliżej.

---

**Listing 6.2.** Prosta galeria obrazów

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">
<title>Galeria obrazów</title>

```

```

</head>
<body>
<?php
$imgDir = "./images";

//odczytanie parametru
if(isset($_GET['imgid'])){
    $imgId = $_GET['imgid'];
}
else{
    $imgId = 0;
}

//odczytanie zawartości katalogu
$dir = scandir($imgDir);
array_shift($dir);
array_shift($dir);

$count = count($dir);

//sprawdzenie poprawności parametru
if($imgId < 0 || $imgId >= $count || !is_numeric($imgId)){
    $imgId = 0;
}

//ustalenie nazwy bieżącego obrazu oraz
//identyfikatorów obrazów dla odnośników
$imgName = $dir[$imgId];
$first = 0;
$last = $count - 1;
if($imgId < $count - 1){
    $next = $imgId + 1;
}
else{
    $next = $count - 1;
}

if($imgId > 0){
    $prev = $imgId - 1;
}
else{
    $prev = 0;
}
?>
<div>
<div id='obraz' style='text-align:center'>
<?php
    echo "<img src=\"\$imgDir/$imgName\" alt=\"\$imgName\" />";
?>
</div>
<div id='opis' style='text-align:center'>
<?php
    $imgId++;
    echo "Obraz $imgName ($imgId z $count)";
?>
</div>
<div id='nawigacja' style='text-align:center'>

```

```
<?php
echo "<a href=\"galeria.php?imgid=$first\">Pierwszy</a> ";
echo "<a href=\"galeria.php?imgid=$prev\">Poprzedni</a> ";
echo "<a href=\"galeria.php?imgid=$next\">Następny</a> ";
echo "<a href=\"galeria.php?imgid=$last\">Ostatni</a> ";
?>
</div>
</div>
</body>
</html>
```

Kod rozpoczyna się od ustawienia zmiennej `$imgDir` wskazującej katalog, w którym będą umieszczane obrazy (musi to być katalog dostępny przez serwer WWW). W tym przypadku jest to podkatalog `images` katalogu, w którym będzie umieszczony skrypt *galeria.php*. Następnie za pomocą instrukcji warunkowej `if` sprawdzane jest, czy do skryptu został przekazany parametr `imgid`, czyli czy w tablicy `$_GET` jest klucz o nazwie `imgid`. Jeśli tak, jego wartość jest odczytywana, konwertowana za pomocą funkcji `intval` do typu `integer` (por. lekcja 4. z rozdziału 2.) i przypisywana zmiennej `$imgId`, w przeciwnym razie wartość zmiennej `$imgId` jest ustawiana na 0.

Po wykonaniu tych wstępnych czynności za pomocą funkcji `scandir` odczytywana jest zawartość katalogu wskazywanego przez `$imgDir` i przypisywana zmiennej `$dir`. Jak wiadomo z lekcji 10. (rozdział 3.), wynikiem działania `scandir` jest tablica zawierająca wszystkie elementy wskazanego katalogu (dlatego też powinny się w nim znajdować wyłącznie pliki graficzne galerii) posortowane w kolejności alfabetycznej. Pierwsze dwa elementy tej tablicy to wskazanie do katalogu bieżącego (`.`) oraz nadrzędnego (`..`)<sup>2</sup>, dlatego też są one usuwane za pomocą dwóch wywołań funkcji `array_shift` (funkcja `array_shift` usuwa z tablicy pierwszą komórkę, czyli pierwszy indeks).

Następnie za pomocą wywołania funkcji `count` pobierana jest i zapisywana w zmiennej `$count` całkowita liczba obrazów, a za pomocą instrukcji `if` następuje sprawdzenie poprawności parametru zapisanego w zmiennej `$imgId`. Nie można bowiem zakładać, że do skryptu została przekazana na pewno poprawna wartość. Dlatego też jest sprawdzane, czy wartość `$imgId` jest:

- ♦ mniejsza od 0,
- ♦ większa od maksymalnej możliwej wartości (`$count - 1`),

Jeśli jeden z tych warunków jest prawdziwy (czyli kiedy zostanie stwierdzona nieprawidłowość przekazanych danych), zmiennej `$imgId` jest przypisywana wartość 0, co oznacza, że zostanie wyświetlony pierwszy obraz.

Po ostatecznym ustaleniu identyfikatora wyświetlanego obrazu są ustawiane wartości zmiennych, które w dalszej części posłużą jako parametry dla znaczników HTML. Są to:

- ♦ `$imgName` — nazwa pliku graficznego pobrana z tablicy `$dir`,
- ♦ `$first` — identyfikator pierwszego obrazu (czyli 0),

---

<sup>2</sup> Wyjątkiem jest sytuacja, gdy parametrem przekazanym funkcji `scandir` jest nazwa katalogu głównego.

- ◆ `$last` — identyfikator ostatniego obrazu, czyli liczba elementów tablicy `$dir` pomniejszona o 1 (`$dir - 1`),
- ◆ `$next` — identyfikator następnego obrazu, czyli wartość aktualnego identyfikatora powiększona o 1 (`$imgId + 1`),
- ◆ `$prev` — identyfikator poprzedniego obrazu, czyli wartość aktualnego identyfikatora pomniejszona o 1 (`$imgId - 1`).

Należy przy tym zwrócić uwagę, że jeśli aktualny identyfikator ma wartość 0, to wartością `$prev` powinno być 0 (nie ma bowiem wcześniejszego obrazu) — i analogicznie jeśli aktualny identyfikator ma wartość równą identyfikatorowi ostatniego obrazu, to wartością `$next` powinien być właśnie on (nie ma bowiem następnego obrazu).

Wszystkie zebrane w opisany sposób dane pozwalają na wygenerowanie znacznika `<img>` w postaci:

```

```

np.:

```

```

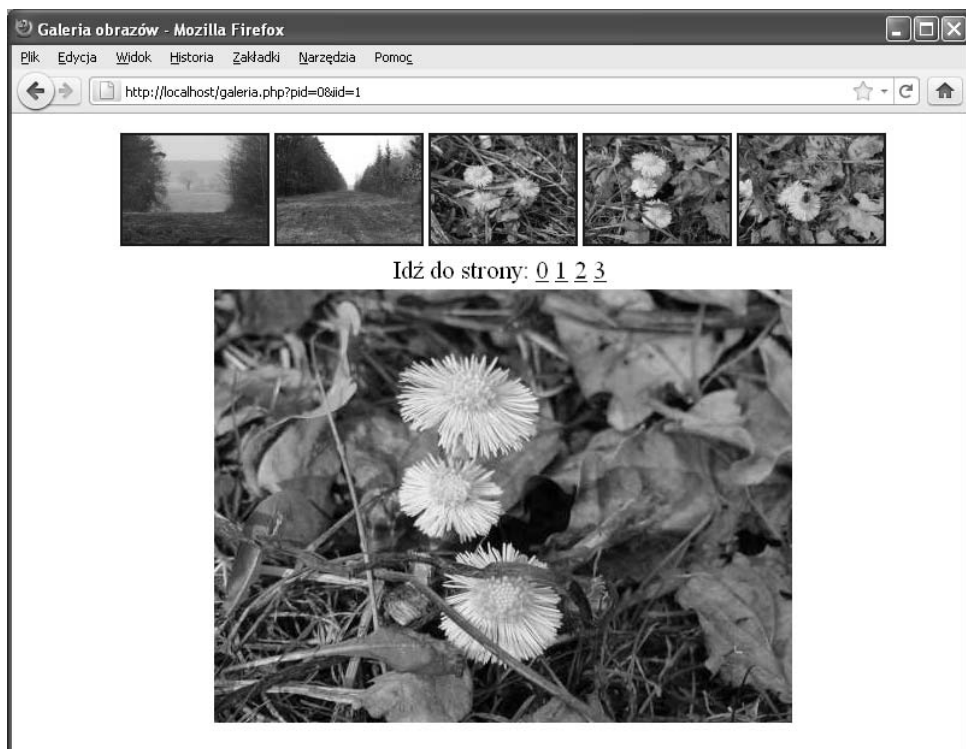
oraz odnośników:

- ◆ `<a href="galeria.php?imgid=$first">Pierwszy</a>` — do obrazu pierwszego,
- ◆ `<a href="galeria.php?imgid=$prev">Poprzedni</a>` — do obrazu poprzedniego,
- ◆ `<a href="galeria.php?imgid=$next">Następny</a>` — do obrazu następnego,
- ◆ `<a href="galeria.php?imgid=$last">Ostatni</a>` — do obrazu ostatniego,
- ◆ oraz pozostałych danych: nazwy pliku graficznego (`$imgName`), numeru aktualnego obrazu (`$imgId + 1`), całkowitej liczby obrazów (`$count`).

## Galeria z miniaturami obrazów

Nieco bardziej złożonym przykładem jest galeria zawierająca miniatury obrazów (taką jak zaprezentowana na rysunku 6.3). U góry strony wyświetlana jest pewna liczba miniatur (zdefiniowana w kodzie skryptu). Kliknięcie każdej z nich powoduje wyświetlenie w dolnej części strony wybranego obrazu w pełnej rozdzielczości. Pod miniaturkami znajdują się z kolei odnośniki pozwalające na nawigację pomiędzy kolejnymi stronami miniatur. Spróbujmy napisać skrypt tworzący taką galerię.

Zacznijmy od przygotowania obrazów. Pliki graficzne zawierające zdjęcia w pełnej rozdzielczości zapiszemy w podkatalogu o nazwie *images*, natomiast miniatury w podkatalogu *thumbnails*. Miniatury można przygotować w dowolnym programie graficznym. Oba podkatalogi zapiszemy w katalogu, w którym będzie znajdował się skrypt realizujący funkcje galerii. Skrypt nazwiemy *galeria.php*. Oczywiście wszystkie wymienione nazwy można zmieniać według własnych potrzeb, będzie to jednak wymagało uwzględnienia wprowadzonych zmian w kodzie. Należy zwrócić przy tym uwagę,



**Rysunek 6.3.** Galeria zawierająca miniatury obrazów

aby w podkatalogach przeznaczonych do umieszczania plików galerii znajdowały się jedynie pliki graficzne, ponieważ skrypt nie będzie sprawdzał typów plików (weryfikacja zostanie pominięta w celu uproszczenia kodu).

Zastanówmy się teraz, jak ma wyglądać struktura strony zawierającej galerię. Może być ona zrealizowana na kilka sposobów. Najprościej jest użyć kilku warstw (które można dowolnie ostylewać) albo też do formatowania użyć zwykłej tabeli (choć nie jest to obecnie sposób zalecany). Skorzystajmy zatem z trzech warstw. Górna będzie przechowywać obrazy miniatur, środkowa — odnośniki do kolejnych stron z miniaturami, a dolna — wybrany przez użytkownika obraz w pełnej rozdzielczości. Struktura ta będzie miała zatem schematyczną postać:

```

<div id='miniatury' style='text-align:center'>
  <!-- lista miniatur -->
</div>
<div id='nawigacja' style='text-align:center'>
  <!-- odnośniki do kolejnych stron z miniaturami -->
</div>
<div id='obraz' style='text-align:center'>
  <!-- obraz w pełnej rozdzielczości-->
</div>

```

Do skryptu będą przekazywane dwa argumenty:

- ◆ `pid` — (od *page id*) określający numer strony z miniaturkami, która ma zostać wyświetlona w górnej części witryny.
- ◆ `iid` — (od *image id*) określający numer obrazu, który ma zostać wyświetlony w dolnej części witryny.

Miniatury powinny mieć takie same nazwy jak właściwe pliki z grafiką i być według nich ustawione w kolejności alfabetycznej. Ponieważ liczba obrazów w galerii nie będzie z góry ustalana (tak aby można było ją aktualizować jedynie przez zmianę zawartości katalogów z grafiką), niezbędne będzie obliczanie liczby stron z miniaturami, co będzie się odbywało według prostego wzoru:

$$\text{liczba stron} = \text{całkowita liczba miniatur} / \text{liczba miniatur na pojedynczej stronie}$$

Kod skryptu generującego galerię został przedstawiony na listingu 6.3. W celu zaoszczędzenia miejsca zostały w nim pominięte wstępne znaczniki HTML, które są takie same jak w poprzednim przykładzie.

### Listing 6.3. Skrypt generujący galerię z miniaturami obrazów

```

<!-- początek kodu HTML jak w przykładzie z listingu 6.2 -->
<body>
<?php
$imgDir = "./images";
$thumbDir = "./thumbnails";
$thOnPage = 5;

if(isset($_GET['iid']) && isset($_GET['pid'])){
    $iId = $_GET['iid'];
    $pId = $_GET['pid'];
}
else{
    $iId = 0;
    $pId = 0;
}

$dir = scandir($thumbDir);
array_shift($dir);
array_shift($dir);

$count = count($dir);
$pages = ceil($count / $thOnPage);

if($iId < 0 || $iId > $count || $pId < 0 || $pId > $pages){
    $iId = 0;
    $pId = 0;
}
?>

<div id='miniatury' style='text-align:center'>

<?php
for($i = 0; $i < $thOnPage; $i++){
    $imgNo = $pId * $thOnPage + $i;
    if($imgNo >= $count) break;
    $imgName = $dir[$imgNo];

```

```

$imgTag = "<img src=\"$thumbDir/$imgName\" alt=\"$imgName\" />\n";
$aHead = "<a href=\"./galeria.php?pid=$pId&iid=$imgNo\">";
$aFoot = "</a>";
echo "$aHead $imgTag $aFoot\n";
}
?>
</div>
<div id='nawigacja' style='text-align:center'>
Idź do strony:
<?php
for($i = 0; $i < $pages; $i++){
    echo "<a href=\"./galeria.php?pid=$i&iid=$iId\">$i</a>&nbsp;";
}
?>
</div>
<div id='obraz' style='text-align:center'>
<?php
    $imgName = $dir[$iId];
    echo "<img src=\"$imgDir/$imgName\" alt=\"$imgName\" />\n";
?>
</div>
</body>
</html>

```

Kod rozpoczyna się od ustawienia zmiennych, które zawierają: nazwę katalogu z obrazami (`$imgDir`), nazwę katalogu z miniaturami (`$thumbDir`) oraz liczbę miniatur wyświetlanych na pojedynczej stronie (`$thOnPage`). Dzięki temu wszelkie modyfikacje tych parametrów będą mogły być wykonywane w prosty sposób w jednym miejscu. Następnie ma miejsce sprawdzenie, czy do skryptu zostały przekazane parametry `pid` i `iid`. Jeśli tak, ich wartości za pomocą funkcji `intval` są konwertowane na liczby całkowite i przypisywane zmiennym `$pid` i `$iid`, jeśli nie — zmienne te otrzymują wartość 0. Oznacza to, że wywołanie skryptu bez podania parametrów spowoduje, iż zostaną wyświetlone strona miniatur oraz obraz, które będą miały indeks 0.

Kolejne instrukcje to wczytanie i zapisanie w zmiennej `$dir` zawartości katalogu z miniaturami oraz usunięcie dwóch pierwszych komórek tej tablicy, czyli pozbycie się wskazań do katalogu bieżącego i nadrzędnego. Te czynności są wykonywane dokładnie w ten sam sposób jak w poprzednim przykładzie. Po wczytaniu danych następuje obliczenie całkowitej liczby plików (zmienna `$count`) oraz wynikającej z niej liczby stron z miniaturami (zmienna `$pages`). Ponieważ w tym drugim przypadku wynik może być liczbą z częścią ułamkową, jest on zaokrąglany do liczby całkowitej za pomocą funkcji `ceil` (zaokrąglenie w górę).

Gdy znamy już zarówno całkowitą liczbę plików, jak i liczbę stron z miniaturami, a zatem również zakres wartości, jakie mogą przyjmować zmienne `$iid` i `$pid`, dokonujemy weryfikacji danych. Nie można bowiem zakładać, że dane przesłane do skryptu na pewno są poprawne. W przypadku stwierdzenia jakiegokolwiek nieprawidłowości (wartości poza dopuszczalnym zakresem) zmiennym `$iid` i `$pid` zostaną przypisane wartości 0, czyli skrypt zachowa się tak, jakby nie zostały mu przekazane żadne parametry.

Po przeprowadzeniu wszystkich opisanych procedur można już przystąpić do wygenerowania górnego wiersza zawierającego obrazy miniatur. Proces ten odbywa się w pętli

for, która ma tyle przebiegów, ile miniatur ma się znaleźć na stronie. Wewnątrz pętli najpierw jest obliczany indeks aktualnie przetwarzanej miniatury (zmienna \$imgNo), który wynika ze wzoru:

$$\text{indeks obrazu} = \text{indeks wyświetlanej strony} * \text{liczba miniatur na stronie} + \$i$$

Gdyby się okazało, że tak wyliczona wartość jest większa od całkowitej liczby obrazów lub jej równa, pętla zostanie przerwana. Oznacza to bowiem, że przetwarzana jest ostatnia strona z miniaturami, na której liczba miniatur może być mniejsza niż na pozostałych stronach.

Po obliczeniu indeksu aktualnej miniatury z tablicy \$dir jest odczytywana (i zapisywana w zmiennej \$imgName) nazwa odpowiadającego jej pliku. Pozwala to na skonstruowanie znacznika <img> w postaci:

```

```

który jest przypisywany zmiennej \$imgTag. Konstruowany jest również znacznik <a>, który spowoduje, że miniatura będzie jednocześnie odnośnikiem pozwalającym na wyświetlenie odpowiadającego jej obrazu w pełnej rozdzielczości. Znacznik ten przyjmuje postać:

```
<a href="/galeria.php?pid=$pId&iid=$imgNo"> znacznik <img> </a>
```

W kodzie został on rozbity na dwie części — \$aHead i \$aFoot — tak aby w prosty sposób można było skonstruować zawartość wysyłaną do przeglądarki za pomocą instrukcji echo. Przyjmuje ona postać:

```
echo "$aHead $imgTag $aFoot";
```

Należy zwrócić uwagę na sposób zapisania znaku & w postaci &amp;. Jest to niezbędne, aby zachować zgodność z obowiązującymi standardami HTML.

Druga część strony (druga warstwa <div>) musi zawierać odnośniki do wszystkich stron z miniaturami. Jej konstrukcja nie jest skomplikowana. Odnośniki są generowane w pętli for, a każdy z nich ma postać:

```
<a href="/galeria.php?pid=$i&iid=$iId">$i</a>&nbsp;
```

Jak widać, zmienia się tu wartość parametru pid, natomiast wartością iid jest indeks aktualnie wyświetlanego obrazu. Dzięki temu obraz w pełnej rozdzielczości wyświetlany w dolnej komórce tabeli będzie się zmieniał tylko na wyraźne życzenie użytkownika przeglądającego galerię, gdy kliknie on wybraną miniaturę.

Równie prosta jest konstrukcja ostatniego, dolnego wiersza witryny (trzeciej warstwy <div>). Powinien się w nim znaleźć jedynie znacznik <img> powodujący wyświetlenie wybranego przez użytkownika obrazu. Nazwa pliku z tym obrazem jest pobierana z tablicy \$dir (jej indeks jest zapisany w zmiennej \$iId) i przypisywana pomocniczej zmiennej \$imgName. Następnie konstruowany jest znacznik <img> w postaci:

```

```

który jest wysyłany do przeglądarki za pomocą instrukcji echo.



## Przetwarzanie grafiki

### Tworzenie obrazu

W celu utworzenia nowego obrazu w pamięci należy wykorzystać funkcję `imagecreate-truercolor`<sup>3</sup>, której jako argumenty przekazuje się szerokość i wysokość obrazu. Schematyczne wywołanie ma zatem postać:

```
imagecreatetruecolor(szerokość, wysokość)
```

Funkcja zwraca identyfikator obrazu (wartość typu `resource`), który pozwala na dalsze operacje. Gdy obraz utworzony za pomocą `imagecreatetruecolor` nie będzie już potrzebny, powinien zostać usunięty z pamięci za pomocą wywołania `imagedestroy`. Funkcja `imagedestroy` zwraca wartość `true`, jeśli operacja się powiedzie, lub `false` w przeciwnym wypadku. Konstrukcja skryptu (dla przykładowego obrazu o rozmiarach 200×100 pikseli) powinna być więc następująca:

```
<?php
$wysokosc = 100;
$szeroosc = 200;
$img = imagecreatetruecolor($szeroosc, $wysokosc);
//operacje na obrazie
imagedestroy($img);
?>
```

W przypadku gdy chcemy wczytać obraz z pliku graficznego, należy użyć funkcji dedykowanej dla danego formatu graficznego<sup>4</sup>:

- ♦ `imagecreatefromgif` — dla formatu GIF,
- ♦ `imagecreatefromjpeg` — dla formatu JPEG,
- ♦ `imagecreatefrompng` — dla formatu PNG.

Każda z nich przyjmuje jako argument nazwę pliku graficznego (o ile konfiguracja środowiska na to pozwala, może być to też ciąg wskazujący URL obrazu). Wczytanie zawartości przykładowego pliku o nazwie *obraz1.jpg*, który znajduje się w katalogu */var/www/images*, osiągniemy przez wykonanie instrukcji:

```
$img = imagecreatefromjpeg("/var/www/images/obraz1.jpg");
```

a gdy ten obraz znajduje się w katalogu *c:\www\obrazy*, dzięki instrukcji:

```
$img = imagecreatefromjpeg("c:\\www\\obrazy\\obraz1.jpg");
```

lub:

```
$img = imagecreatefromjpeg("c:/www/obrazy/obraz1.jpg");
```

<sup>3</sup> Można także użyć funkcji `imagecreate`, zaleca się jednak korzystanie z `imagecreatetruecolor`.

<sup>4</sup> Dostępne są także funkcje dla mniej popularnych formatów, np. `gd2`, `bmp`, `xpm` itp. Ich opis można znaleźć w dokumentacji PHP.

## Zapisywanie plików graficznych

Obraz utworzony w pamięci (niezależnie od tego, jak ta czynność została wykonana) może być w prosty sposób zapisany do pliku lub wysłany do standardowego wyjścia (np. przeglądarki). W zależności od wybranego formatu należy użyć jednej z dedykowanych funkcji:

- ◆ `imagegif` — dla formatu GIF,
- ◆ `imagejpeg` — dla formatu JPEG,
- ◆ `imagepng` — dla formatu PNG.

Każda z nich przyjmuje dwa argumenty — pierwszy określa obraz, natomiast drugi nazwę pliku. Jeśli parametr określający nazwę pliku zostanie pominięty, obraz jest wysyłany do standardowego wyjścia. Aby na przykład zapisać na dysku plik typu JPEG o nazwie *image.jpg* utworzony z obrazu wskazywanego przez zmienną `$img`, zastosujemy konstrukcję:

```
imagejpeg($img, "image.jpg");
```

aby natomiast wysłać taki obraz do przeglądarki — konstrukcję<sup>5</sup>:

```
imagejpeg($img);
```

Funkcje `imagejpg` oraz `imagepng` mogą też przyjmować trzeci argument określający jakość pliku wynikowego. W przypadku `imagejpg` jest to wartość od 0 do 100, przy czym 0 oznacza największą kompresję i największą utratę jakości (ale najmniejszy rozmiar pliku wynikowego), natomiast 100 — najmniejszą kompresję i najmniejszą utratę jakości (ale największy rozmiar pliku wynikowego). W przypadku `imagepng` jest to wartość od 0 do 9, gdzie 0 oznacza brak kompresji, a 9 największą kompresję<sup>6</sup>.

Jeżeli stosowany jest trzeci argument, a obraz powinien być wysłany do przeglądarki, jako nazwy pliku (drugi argument) należy użyć wartości `null`, np.:

```
imagejpeg($img, null, 85);
```

## Kolory

Wiele funkcji operujących na obrazie wymaga podania w postaci argumentu koloru, który ma być użyty (np. funkcja rysująca linie wymaga podania koloru linii). Aby jednak móc skorzystać z danego koloru, niezbędne jest wcześniejsze jego zaalokowanie, które odbywa się przez wywołanie funkcji `imagecolorallocate`. Jej wywołanie ma postać:

```
imagecolorallocate($obraz, czerwony, zielony, niebieski)
```

gdzie `$obraz` to identyfikator zwrócony przez jedną z funkcji tworzących obraz, natomiast pozostałe argumenty określają poszczególne składowe koloru w formacie RGB. Mogą być one podawane w postaci dziesiętnej lub szesnastkowej. Wybrane kolory

<sup>5</sup> W takim przypadku niezbędne byłoby również wcześniejsze wysłanie odpowiedniego nagłówka HTTP.

<sup>6</sup> Ten argument jest dostępny od PHP w wersji 5.1.2. Począwszy od wersji 5.1.3, jest też możliwe stosowanie czwartego argumentu określającego filtry, jakie mają być zastosowane do obrazu.

i odpowiadające im składowe RGB zostały przedstawione w tabeli 6.1. Funkcja `imagecolorallocate` zwraca identyfikator koloru lub wartość `false` (w wersjach PHP poniżej 5.1.3 — `-1`), jeśli jej działanie nie zakończyło się powodzeniem. Przykładowo, uzyskanie indeksu koloru czerwonego dla obrazu wskazywanego przez zmienną `$img` będzie wymagało wywołania w postaci:

```
$czerwony = imagecolorallocate($img, 255, 0, 0);
```

lub:

```
$czerwony = imagecolorallocate($img, 0xFF, 0, 0);
```

**Tabela 6.1.** Wybrane kolory i odpowiadające im składowe RGB

Kolor	Składowa R	Składowa G	Składowa B
Beżowy	245	245	220
Biały	255	255	255
Błękitny	0	191	255
Brazowy	139	69	19
Czarny	0	0	0
Czerwony	255	0	0
Ciemnoczerwony	192	0	0
Ciemnoniebieski	0	0	139
Ciemnoszary	169	169	169
Ciemnozielony	0	100	0
Fiolet	238	130	238
Koralowy	255	127	80
Niebieski	0	0	255
Oliwkowy	128	128	0
Purpurowy	128	0	128
Srebrny	192	192	192
Stalowiebieski	70	130	180
Szary	128	128	128
Zielony	0	255	0
Żółtozielony	192	255	62
Żółty	255	255	0

Jeśli chcemy wypełnić wybranym kolorem pewien obszar obrazu, można skorzystać z funkcji `imagefill`, której wywołanie ma postać:

```
imagefill($obraz, $wsp_x, $wsp_y, $kolor);
```

gdzie `$obraz` określa obraz, `$wsp_x` i `$wsp_y` — współrzędną punktu, od którego rozpocznie się procedura wypełniania, a `$kolor` — kolor wypełnienia. Aby na przykład w obrazie wskazywanym przez zmienną `$img` wypełnić kolorem niebieskim obszar rozpoczynający się od punktu o współrzędnych (15, 25), należy zastosować instrukcję:

```
$niebieski = imagecolorallocate($img, 0, 0, 0xFF);
imagefill($img, 15, 25, $niebieski);
```

## Informacje o obrazie

Jeśli chcemy otrzymać informacje o rozmiarach znajdującego się w pamięci obrazu, możemy skorzystać z funkcji `imagesx` (z ang. *image size x*) i `imagesy` (z ang. *image size y*). Pierwsza z nich zwraca szerokość, a druga wysokość obrazu. W obu przypadkach jako argument należy podać odnośnik do obrazu. Jeśli zatem utworzyliśmy lub wczytaliśmy z pliku obraz, który jest identyfikowany przez zmienną `$img`, to jego rozmiary możemy uzyskać za pomocą przykładowych wywołań:

```
$szerokosc = imagesx($img);
$wysokosc = imagesy($img);
```

Równie przydatną funkcją jest `getimagesize`, która (nieco wbrew swojej nazwie) zwraca wiele przydatnych informacji, a nie tylko rozmiary obrazu. Jej wywołanie ma postać:

```
getimagesize(nazwa_pliku[, $tablica]);
```

gdzie *nazwa\_pliku* określa nazwę pliku, z którego dane chcemy odczytać (może to być zarówno plik lokalny, jak i zdalny), a *tablica* — opcjonalną tablicę, w jakiej zostaną zapisane dodatkowe informacje (w obecnej wersji PHP zapisywane są w niej jedynie niektóre znaczniki APP z plików JPG).

Wartością zwracaną przez `getimagesize` jest tablica o następującej zawartości:

- ◆ indeks 0 — szerokość obrazu w pikselach,
- ◆ indeks 1 — wysokość obrazu w pikselach,
- ◆ indeks 2 — określenie typu pliku (dostępne wartości zostały zaprezentowane w tabeli 6.2),
- ◆ indeks 3 — ciąg znaków zapisany w postaci `height="wysokość" width="szerokość"`, który określa rozmiary obrazu,
- ◆ klucz `mime` — ciąg znaków określający typ *mime* pliku.
- ◆ klucz `channels` — liczba kanałów (3 dla RGB, 4 dla CMYK).
- ◆ klucz `bits` — liczba bitów, na których zapisywany jest kolor (z czego bezpośrednio wynika maksymalna możliwa liczba kolorów w obrazie).

## Generowanie grafiki

Na obrazie utworzonym lub wczytanym w sposób opisany w sekcji „Tworzenie obrazu” można za pomocą odpowiednich funkcji wykonywać różne operacje graficzne. Na kolejnych stronach omówione zostanie kilka z nich. Funkcje te pozwalają m.in. na rysowanie linii, łuków, prostokątów, wielokątów i elips.

### Rysowanie linii

Do rysowania linii służy funkcja `imageline`, której wywołanie ma postać:

```
imageline($obraz, xp, yp, xk, yk, kolor)
```

**Tabela 6.2.** Typy plików rozpoznawane przez funkcję *getimagesize*

Typ	Wartość
GIF	1
JPEG	2
PNG	3
SWF	4
PSD	5
BMP	6
TIFF_II <sup>7</sup>	7
TIFF_MM <sup>8</sup>	8
JPC	9
JPEG_2000	9
JP2	10
JPX	11
JB2	12
SWC	13
IFF	14
WBMP	15
XBM	16
ICO <sup>9</sup>	17

gdzie:

- ♦ *\$obraz* — to określenie obrazu,
- ♦ *xp* — współrzędna x początku linii,
- ♦ *yp* — współrzędna y początku linii,
- ♦ *xk* — współrzędna x końca linii,
- ♦ *yk* — współrzędna y końca linii,
- ♦ *kolor* — określenie koloru.

Należy przy tym pamiętać, że współrzędne lewego górnego rogu to (0, 0). Przykładowo, aby utworzyć obraz o białym tle i o rozdzielczości 100×30 pikseli, który by zawierał dwie przekątne (pierwszą koloru zielonego, a drugą czerwonego), i zapisać go na dysku pod nazwą *obraz1.jpg*, należy wykonać kod widoczny na listingu 6.4.

<sup>7</sup> Kolejność bajtów zgodna z formatem Intela (z ang. *intel byte order*).

<sup>8</sup> Kolejność bajtów zgodna z formatem Motoroli (z ang. *motorola byte order*).

<sup>9</sup> Począwszy od PHP w wersji 5.3.0.

**Listing 6.4.** *Rysowanie przekątnych*

```
<?php
$img = imagecreatetruecolor(100, 30);
$bialy = imagecolorallocate($img, 255, 255, 255);
$zielony = imagecolorallocate($img, 0, 255, 0);
$czerwony = imagecolorallocate($img, 255, 0, 0);

imagefill($img, 0, 0, $bialy);

imageline($img, 0, 0, 99, 29, $czerwony);
imageline($img, 0, 29, 99, 0, $zielony);

imagejpeg($img, "obraz1.jpg");
imagedestroy($img);
?>
```

Obraz jest tworzony za pomocą funkcji `imagecreatetruecolor`. Ta funkcja zwraca zasób będący odwołaniem do obrazu (czy też identyfikatorem obrazu). A zatem w dalszej części skryptu obraz będzie reprezentowany przez zmienną `$img`. W uproszczeniu często mówi się po prostu o obrazie `$img`.

Następnie za pomocą funkcji `imagecolorallocate` alokowane są trzy kolory: biały, zielony i czerwony. Wyniki działania funkcji (identyfikatory kolorów) są przypisywane zmiennym `$bialy`, `$zielony` i `$czerwony`. Obszar obrazu jest wypełniany kolorem białym dzięki funkcji `imagefill`, a przekątne są rysowane przez funkcję `imageline`. Na zakończenie obraz jest zapisywany do pliku o nazwie `obraz1.jpg` (odpowiada za to funkcja `imagejpeg`) oraz usuwany z pamięci (odpowiada za to funkcja `imagedestroy`).

## Rysowanie prostokątów

Do rysowania prostokątów służą funkcje `imagerectangle` i `imagefilledrectangle`, których wywołanie ma postać:

```
imagerectangle($obraz, xp, yp, xk, yk, color)
imagefilledrectangle($obraz, xp, yp, xk, yk, color)
```

gdzie:

- ◆ `$obraz` — to określenie obrazu,
- ◆ `xp` — współrzędna x lewego górnego rogu,
- ◆ `yp` — współrzędna y lewego górnego rogu,
- ◆ `xk` — współrzędna x prawego dolnego rogu,
- ◆ `yk` — współrzędna y prawego dolnego rogu,
- ◆ `color` — określenie koloru.

Pierwsza funkcja tworzy sam kontur prostokąta, a druga prostokąt wypełniony kolorem. Aby zatem utworzyć obraz o białym tle i o rozdzielczości 100×100 pikseli, który by zawierał cztery ułożone w szachownicę prostokąty w kolorach niebieskim i zielonym (w tym dwa wypełnione własnym kolorem i dwa wypełnione kolorem tła), oraz zapisać go na dysku pod nazwą `obraz1.jpg`, należy wykonać kod widoczny na listingu 6.5.

**Listing 6.5.** Tworzenie prostokątów

```
<?php
$img = imagecreatetruecolor(100, 100);
$bialy = imagecolorallocate($img, 255, 255, 255);
$zielony = imagecolorallocate($img, 0, 255, 0);
$niebieski = imagecolorallocate($img, 0, 0, 255);

imagefill($img, 0, 0, $bialy);

imagefilledrectangle($img, 10, 10, 50, 50, $niebieski);
imagefilledrectangle($img, 50, 50, 90, 90, $niebieski);
imagerectangle($img, 10, 50, 50, 90, $zielony);
imagerectangle($img, 50, 10, 90, 50, $zielony);

imagejpeg($img, "obraz1.jpg");
imagedestroy($img);
?>
```

**Rysowanie wielokątów**

Do rysowania wielokątów służą funkcje `imagepolygon` i `imagefilledpolygon`, których wywołanie ma postać:

```
imagepolygon($obraz, $punkty, $ile, $kolor)
imagefilledpolygon($obraz, $punkty, $ile, $kolor)
```

gdzie:

- ♦ *\$obraz* — to określenie obrazu,
- ♦ *\$punkty* — tablica zawierająca współrzędne kolejnych punktów,
- ♦ *\$ile* — liczba wierzchołków,
- ♦ *\$kolor* — określenie koloru.

Tablica *punkty* musi być zbudowana w taki sposób, że klucze 0 i 1 zawierają współrzędne *x* i *y* pierwszego wierzchołka, klucze 2 i 3 — drugiego wierzchołka, 4 i 5 — trzeciego wierzchołka itd. Funkcja `imagepolygon` rysuje sam kontur wielokąta, natomiast `imagefilledpolygon` wielokąt wypełniony wskazanym kolorem. Aby na przykład uzyskać obraz o białym tle i o rozdzielczości 320×200 pikseli, który by zawierał sześciokąt wypełniony kolorem żółtym, należy wykorzystać kod widoczny na listingu 6.6.

**Listing 6.6.** Tworzenie sześciokąta wypełnionego kolorem żółtym

```
<?php
$img = imagecreatetruecolor(320, 200);
$bialy = imagecolorallocate($img, 255, 255, 255);
$zolyty = imagecolorallocate($img, 255, 255, 0);

imagefill($img, 0, 0, $bialy);

$stab = array(80, 100, 120, 20, 200, 20, 240, 100, 200, 180, 120, 180);

imagefilledpolygon($img, $stab, 6, $zolyty);
```

```
imagejpeg($img, "obraz1.jpg");
imagedestroy($img);
?>
```

## Rysowanie elips

Do rysowania elips służą funkcje `imageellipse` i `imagefilledellipse`, których wywołanie ma postać:

```
imageellipse($obraz, xc, yc, xw, xh, kolor)
imagefilledellipse($obraz, xc, yc, xw, xh, kolor)
```

gdzie:

- ◆ *\$obraz* — to określenie obrazu,
- ◆ *xc* — współrzędna x środka elipsy,
- ◆ *yc* — współrzędna y środka elipsy,
- ◆ *xw* — szerokość elipsy (średnica pozioma),
- ◆ *xh* — wysokość elipsy (średnica pionowa),
- ◆ *kolor* — określenie koloru.

Funkcja `imageellipse` rysuje sam kontur, natomiast `imagefilledellipse` elipsę wypełnioną wskazanym kolorem. Aby na przykład uzyskać obraz o białym tle i o rozdzielczości 320×200 pikseli, który by zawierał czarny okrąg oraz elipsę wypełnioną kolorem niebieskim, należy wykorzystać kod widoczny na listingu 6.7.

### Listing 6.7. Rysowanie elips

```
<?php
$img = imagecreatetruecolor(320, 200);
$biały = imagecolorallocate($img, 255, 255, 255);
$czarny = imagecolorallocate($img, 0, 0, 0);
$niebieski = imagecolorallocate($img, 0, 0, 255);

imagefill($img, 0, 0, $biały);

imagefilledellipse($img, 100, 100, 100, 30, $niebieski);
imageellipse($img, 220, 100, 80, 80, $czarny);

imagejpeg($img, "obraz1.jpg");
imagedestroy($img);
?>
```

## Rysowanie wycinków elips

Do rysowania wycinków elips służą funkcje `imagearc` i `imagefilledarc`, których wywołanie ma postać:

```
imagearc($obraz, xc, yc, xw, xh, k1, k2, kolor)
imagefilledarc($obraz, xc, yc, xw, xh, k1, k2, kolor, styl)
```

gdzie:



- ♦ `$obraz` — to określenie obrazu,
- ♦ `xc` — współrzędna x środka elipsy,
- ♦ `yc` — współrzędna y środka elipsy,
- ♦ `xw` — szerokość elipsy (średnica pozioma),
- ♦ `xh` — wysokość elipsy (średnica pionowa),
- ♦ `k1` — kąt (w stopniach) określający linię początkową,
- ♦ `k2` — kąt (w stopniach) określający linię końcową,
- ♦ `color` — określenie koloru,
- ♦ `styl` — styl wypełnienia.

Pierwsza z nich rysuje sam łuk, druga wycinek elipsy zgodnie ze stylem przekazanym w argumentcie `styl`. Argument ten składa się z następujących stałych, które można łączyć przez sumę bitową (operator `|`):

- ♦ `IMG_ARC_PIE` — wycinek, którego końce są połączone łukiem, standardowo wypełniony zadany kolor.
- ♦ `IMG_ARC_CHORD` — wycinek, którego końce połączone są linią prostą, standardowo wypełniony zadany kolor.
- ♦ `IMG_ARC_NOFILL` — wycinek nie będzie wypełniany kolorem.
- ♦ `IMG_ARC_EDGED` — w połączeniu z `IMG_ARC_NOFILL` powoduje, że zostanie wykreślony pełny kontur wycinka.

Opcje `IMG_ARC_PIE` i `IMG_ARC_CHORD` wzajemnie się wykluczają, czyli nie mogą być użyte jednocześnie. Wygenerowanie obrazu widocznego na rysunku 6.4 osiągniemy zatem, stosując kod zaprezentowany na listingu 6.8.

**Rysunek 6.4.**  
Wykorzystanie różnych opcji funkcji `imagefilledarc`



**Listing 6.8.** Różne sposoby rysowania wycinków elips

```
<?php
$img = imagecreatetruecolor(400, 200);
$bialy = imagecolorallocate($img, 255, 255, 255);
$czerwony = imagecolorallocate($img, 255, 0, 0);
$niebieski = imagecolorallocate($img, 0, 0, 255);

imagefill($img, 0, 0, $bialy);

imagearc($img, 110, 100, 80, 80, 0, 90, $czerwony);
imagearc($img, 90, 100, 80, 80, 90, 180, $czerwony);
imagearc($img, 90, 90, 80, 80, 180, 270, $czerwony);
imagearc($img, 110, 90, 80, 80, 270, 360, $czerwony);
```

```

imagefilledarc($img, 310, 100, 80, 80, 0, 90, $niebieski, IMG_ARC_PIE);
imagefilledarc($img, 290, 100, 80, 80, 90, 180, $niebieski, IMG_ARC_CHORD);
imagefilledarc($img, 290, 90, 80, 80, 180, 270, $niebieski, IMG_ARC_NOFILL);
imagefilledarc($img, 310, 90, 80, 80, 270, 360, $niebieski, IMG_ARC_NOFILL |
    IMG_ARC_EDGED);

imagejpeg($img, "obraz1.jpg");
imagedestroy($img);
?>

```

## Przetwarzanie obrazów

### Nakładanie filtrów

Wśród funkcji przetwarzających obrazy znajduje się `imagefilter`, która nakłada na nie jeden z dostępnych filtrów. Jej wywołanie ma postać:

```
imagefilter($obraz, filtr[, arg1[, arg2[, arg3]]])
```

Argument *\$obraz* określa obraz poddawany zmianie, *filtr* — rodzaj filtra (dostępne wartości zostały podane w tabeli 6.3), natomiast *arg1*, *arg2*, *arg3* i *arg4* to parametry niezbędne do działania niektórych filtrów. Funkcja zwraca wartość `true`, jeśli jej działanie zakończyło się sukcesem, lub `false` w przeciwnym razie.

**Tabela 6.3.** Filtry dostępne dla funkcji `imagefilter`

Nazwa filtra	Opis
IMG_FILTER_NEGATE	Inwersja kolorów.
IMG_FILTER_GRAYSCALE	Konwersja obrazu do odcieni szarości.
IMG_FILTER_BRIGHTNESS	Zmiana poziomu jasności obrazu. Do jego ustalenia należy wykorzystać argument <i>arg1</i> .
IMG_FILTER_CONTRAST	Zmiana kontrastu obrazu. Do ustalenia kontrastu należy wykorzystać argument <i>arg1</i> .
IMG_FILTER_COLORIZE	Działanie podobne do IMG_FILTER_GRAYSCALE z tą różnicą, że istnieje możliwość ustalenia koloru bazowego. Kolor ten należy podać w formacie RGB, gdzie R to wartość <i>arg1</i> , G — wartość <i>arg2</i> , B — wartość <i>arg3</i> , a kanał alfa (przezroczystość) — <i>arg4</i> (dostępny od PHP 5.2.5).
IMG_FILTER_EDGEDETECT	Uwypuklenie krawędzi w obrazie.
IMG_FILTER_EMOSS	Wytłoczenie obrazu.
IMG_FILTER_GAUSSIAN_BLUR	Rozmycie obrazu metodą Gaussa.
IMG_FILTER_SELECTIVE_BLUR	Rozmycie obrazu.
IMG_FILTER_MEAN_REMOVAL	Uwypuklenie krawędzi.
IMG_FILTER_SMOOTH	Wygładzenie (zmiękczenie, rozmycie) obrazu. Parametr <i>arg1</i> pozwala na określenie poziomu intensywności efektu.
IMG_FILTER_PIXELATE	Pikselizacja obrazu. Argument <i>arg1</i> określa rozmiar bloku, natomiast <i>arg2</i> ustawiony na <code>true</code> (domyślnie <code>false</code> ) włącza przetwarzanie zaawansowane (większe wygładzenie obrazu wynikowego).

Można zatem napisać przykładowy skrypt, który podczas wywoływania w wierszu poleceń będzie otrzymywał nazwę pliku graficznego w formacie JPG oraz wartość całkowitą, a także wykona zmianę poziomu kontrastu tego pliku. Działający w ten sposób kod został zaprezentowany na listingu 6.9.

---

**Listing 6.9.** *Zmiana poziomu kontrastu*

---

```
<?php
if($argc < 3){
    exit("Wywołanie skryptu: php konwertuj.php nazwa_pliku poziom_kontrastu\n");
}

if(!$img = imagecreatefromjpeg($argv[1])){
    exit("Nie udało się wczytać pliku {$argv[1]}\n");
}

if(!is_numeric($argv[2])){
    exit("Poziom kontrastu musi być liczbą całkowitą!\n");
}

if(!imagefilter($img, IMG_FILTER_CONTRAST, $argv[2])){
    exit("Nie udało się zmienić poziomu kontrastu w pliku {$argv[1]}.");
}

if(!imagejpeg($img, $argv[1])){
    exit("Wystąpił błąd podczas zapisu pliku {$argv[1]}.");
}

echo "Operacja zmiany kontrastu zakończona sukcesem!";
imagedestroy($img);
?>
```

---

## Skalowanie

W celu przeskalowania obrazu do zadanych rozmiarów można użyć funkcji `imagecopyresized` lub `imagecopyresampled`. W rzeczywistości pobierają one określony parametrami wycinek z obrazu źródłowego i wstawiają go w miejsce obrazu docelowego (również określone parametrami). Jeśli wyznaczone obszary obrazu źródłowego i docelowego nie będą takie same, nastąpi odpowiednie przeskalowanie. Różnica pomiędzy `imagecopyresized` a `imagecopyresampled` jest taka, że druga z nich podczas skalowania dokonuje interpolacji pikseli, dzięki czemu uzyskuje się dokładniejszy efekt. Obie funkcje przyjmują identyczne zestawy argumentów, a ich wywołania mają postać:

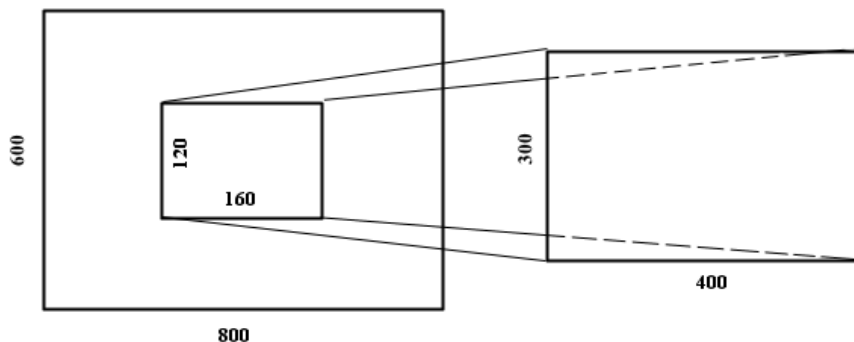
```
imagecopyresized($obraz_docelowy, $obraz_źródłowy, xd, yd, xs, ys, wd, hd, ws, hs);
imagecopyresampled($obraz_docelowy, $obraz_źródłowy, xd, yd, xs, ys, wd, hd, ws, hs);
```

gdzie:

- ♦ `$obraz_docelowy` — określa obraz docelowy,
- ♦ `$obraz_źródłowy` — określa obraz źródłowy,
- ♦ `xd` — współrzędna x lewego górnego rogu kopiowanego obszaru w obrazie docelowym,

- ◆  $yd$  — współrzędna y lewego górnego rogu kopiowanego obszaru w obrazie docelowym,
- ◆  $xs$  — współrzędna x lewego górnego rogu kopiowanego obszaru w obrazie źródłowym,
- ◆  $ys$  — współrzędna y lewego górnego rogu kopiowanego obszaru w obrazie źródłowym,
- ◆  $wd$  — szerokość kopiowanego obszaru w obrazie docelowym,
- ◆  $hd$  — wysokość kopiowanego obszaru w obrazie docelowym,
- ◆  $ws$  — szerokość kopiowanego obszaru w obrazie źródłowym,
- ◆  $hs$  — wysokość kopiowanego obszaru w obrazie źródłowym.

Jak by to wyglądało w praktyce? Załóżmy na przykład, że mamy plik o nazwie *obraz.jpg* zawierający obraz o rozdzielczości 800×600 pikseli. Chcielibyśmy powiększyć jego środkowy fragment o wielkości 160×120 pikseli do rozdzielczości 400×300 (chodzi zatem o wykonanie operacji przedstawionej schematycznie na rysunku 6.5) i tak przetworzoną część zapisać w pliku o nazwie *obraz2.jpg*.



Rysunek 6.5. Schemat skalowania obrazu

Kolejność wykonywanych czynności będzie zatem następująca:

1. Wczytanie obrazu z pliku *obraz1.jpg* za pomocą funkcji `imagecreatefromjpeg`.
2. Utworzenie nowego obrazu za pomocą funkcji `imagecreatetruecolor`.
3. Przeskalowanie wycinka obrazu oryginalnego za pomocą funkcji `imagecopyresampled`.
4. Zapisanie przeskalowanego obrazu do pliku *obraz2.jpg* za pomocą funkcji `imagejpeg`.
5. Usunięcie obrazów z pamięci za pomocą funkcji `imagedestroy`.

Pozostaje jeszcze ustalenie argumentów funkcji `imagecopyresampled`. Nie jest to trudne. Ponieważ przeskalowany fragment będzie zajmował cały obszar obrazu docelowego, współrzędne lewego górnego wierzchołka to 0,0, a rozdzielczość to 400×300

pikseli. Długość i szerokość powiększanego fragmentu obrazu źródłowego też jest nam znana — to 160×120. Pozostaje więc ustalenie współrzędnych lewego górnego wierzchołka tego obszaru. Można to wyliczyć ze wzorów:

$$x = (\text{szerokość obrazu źródłowego} - \text{szerokość powiększanego wycinka}) / 2$$

$$y = (\text{wysokość obrazu źródłowego} - \text{wysokość powiększanego wycinka}) / 2$$

co w tym przypadku daje  $x = 320$  i  $y = 240$ . Kod skryptu wykonującego przedstawione zadania został zaprezentowany na listingu 6.10.

**Listing 6.10.** Skrypt skalujący środkową część obrazu do zadanej rozdzielczości

```
<?php
if(!$img_src = imagecreatefromjpeg("obraz1.jpg"))
    exit("Nie udało się wczytać pliku obraz1.jpg.\n");

if(!$img_dest = imagecreatetruecolor(400, 300))
    exit("Nie udało się utworzyć nowego obrazu.\n");

if(!imagecopyresampled($img_dest, $img_src, 1, 1, 320, 240, 400, 300, 160, 120))
    exit("Nie udało się operacja skalowania.\n");

if(!imagejpeg($img_dest, "obraz2.jpg"))
    exit("Wystąpił błąd podczas zapisu pliku obraz2.jpg.\n");

echo "Operacja skalowania zakończona sukcesem!\n";

imagedestroy($img_src);
imagedestroy($img_dest);
?>
```

## Obracanie

Do obracania obrazów służy funkcja `imagerotate`, której wywołanie ma postać:

```
imagerotate($obraz_źródłowy, kąt_obrotu, kolor_tła, ignoruj_przezroczyste);
```

Obraca ona obraz wskazywany przez `$obraz_źródłowy` o kąt `kąt_obrotu`, wypełniając ewentualne powstałe przy tym puste obszary kolorem `kolor_tła`. Ustawienie argumentu `ignoruj_przezroczyste` na wartość inną niż 0 powoduje, że fragmenty z oznaczoną przezroczystością będą ignorowane (wartość domyślna to 0; argument wprowadzony w PHP 5.1.0). Funkcja zwraca przetworzony obraz. Aby zatem obrócić obraz wskazywany przez zmienną `$img` o 45 stopni (wypełniając puste obszary kolorem białym), a obraz wynikowy przypisać zmiennej `$obraz`, należałoby wykonać instrukcje:

```
$biały = imagecolorallocate($img, 255, 255, 255);
$obraz = imagerotate($img, 45, $biały);
```

## Ćwiczenia do samodzielnego wykonania

Ćwiczenie 20.1. Zmodyfikuj skrypt generujący galerię z listingu 6.2 w taki sposób, aby zachowywał się prawidłowo, gdy w katalogu przeznaczonym do przechowywania obrazów:

- a) nie ma żadnego pliku,
- b) zapisane są pliki o rozszerzeniach innych niż *jpg*, *gif*, *png*,
- c) znajdują się podkatalogi.

Ćwiczenie 20.2. Napisz taki skrypt realizujący galerię obrazów, żeby każdy obraz mógł zawierać podpis pobierany z pliku tekstowego o nazwie zgodnej z nazwą pliku z obrazem, ale o rozszerzeniu *txt*.

Ćwiczenie 20.3. Napisz skrypt, który będzie wykonywał skalowanie obrazu z pliku graficznego typu JPG (lub innego) do zadanej rozdzielczości. Nazwa pliku oraz rozdzielczość powinny być podawane w postaci argumentu.

Ćwiczenie 20.4. Napisz skrypt skalujący wszystkie obrazy typu JPG (lub innego) zapisane w danym katalogu o zadaną wartość procentową.

Ćwiczenie 20.5. Napisz skrypt wykonujący konwersje pomiędzy różnymi formatami plików graficznych.

# Skorowidz

- , 69  
-- , 76  
!, 389  
!= , 77, 80, 387  
!== , 77, 80  
" , 66  
# , 48  
\*/ , 47  
\$ , 57, 62, 66  
\$\_COOKIE , 85  
\$\_ENV , 85  
\$\_FILES , 85  
\$\_GET , 85  
\$\_POST , 85  
\$\_REQUEST , 85  
\$\_SERVER , 84  
\$\_SERVER (tablica) , 339  
\$\_SESSION , 85  
\$dbo = null; , 412  
\$dbo->query("Treść zapytania") 413  
\$GLOBALS , 84  
\$host , 349  
\$HTTP\_COOKIE\_VARS , 85  
\$HTTP\_ENV\_VARS , 85  
\$HTTP\_GET\_VARS , 85  
\$HTTP\_POST\_FILES , 85  
\$HTTP\_POST\_VARS , 85  
\$HTTP\_SERVER\_VARS , 84  
\$pass , 349  
\$plik\_lokalny , 349  
\$plik\_zdalny , 349  
\$user , 349  
% , 69  
%% , 149  
%= , 76  
%> , 46  
%kod , 149  
& , 71, 124  
&& , 388  
&= , 76  
\* , 69  
\*= , 76  
.. , 83  
., 142  
.= , 76  
/ , 69  
/\* , 47  
// , 48  
/= , 76  
?> , 42, 45, 46  
@ , 81  
\ , 66  
^ , 71  
^= , 76  
\_\_autoload , 284, 285  
\_\_construct , 281  
| , 71  
|| , 388  
|= , 76  
~ , 71  
+ , 69, 79  
++ , 76  
+= , 76  
< , 77, 387  
<!DOCTYPE> , 42  
<% , 46  
</p> , 44  
</script> , 46  
<? , 46  
<?php , 42, 45  
<< , 71  
<<< , 60  
<<= , 76  
<= , 77, 388  
<=> , 387  
<> , 77, 80, 387

<body>, 42  
 <h2>, 50  
 <head>, 42  
 <meta>, 42  
 <p>, 42  
 <script>, 46  
 <title>, 42  
 =, 76, 387  
 -=, 76  
 ==, 77, 80  
 ===, 77, 80  
 >, 77, 388  
 ->, 279  
 >=, 77, 388  
 >>, 71  
 >>=, 76  
 0, 130

## A

a, 187  
 a+, 187  
 access modifiers, 297  
 adres IP, 342, 433
 

- blokada, 342
- kontrola dostępu, 343
- różne wersje strony, 343
- zliczanie (odwiedziny), 286

 adresy lokalne, 14  
 akapit tekstowy (wyświetlanie w przeglądarce), 146  
 aktualny katalog, 178  
 algorytm sortowania QuickSort, 168  
 ALTER TABLE, 374  
 alternatywa logiczna, 74  
 am, 127  
 AM, 127  
 ampersand, 81, 124  
 AND, 71, 74, 388  
 Apache, 8, 410
 

- instalacja (Linux Fedora), 14
- instalacja (Linux Ubuntu), 20, 21
- instalacja (Windows), 22
- Service Monitor, 25
- uruchamianie (Linux Fedora), 17

 apostrof, 60, 64  
 argc, 340  
 argument
 

- domyślny, 125
- funkcji, 115
- przekazywanie przez referencje, 123
- przekazywanie przez wartość, 123

 argv, 339  
 array, 158, 161

array\_serach, 235  
 arytmetyczny operator dodawania, 142  
 AS, 386  
 asort, 165  
 AUTH\_TYPE, 341  
 AUTO\_INCREMENT, 373  
 autoglobal, 123  
 automatyczne generowanie listy plików, 237

## B

backslash, 66  
 baza danych
 

- kończenie połączenia, 395
- nawiązanie połączenia, 394
- wybór, 370, 396

 BETWEEN N AND M, 388  
 białe znaki (usuwanie), 150  
 biblioteka
 

- GD, 315
- graficzna, 316

 BINARY, 380  
 bitowa różnica symetryczna, 72  
 BLOB, 381  
 błąd (sygnalizacja wystąpienia), 305  
 boolean, 58  
 break, 105  
 button (formularz), 216  
 by
 

- reference, 123
- value, 123

## C

C, 187  
 C+, 187  
 całkowitoliczbowy, 377  
 carriage return, 66, 151  
 catch, 306  
 CHAR, 380  
 chdir, 177  
 checkbox (formularz), 216  
 checkdate, 139  
 checkFileName, 235  
 checkPass, 273, 427  
 ciało funkcji, 114  
 ciąg
 

- formatowanie, 146
- porównanie, 154
- przeszukiwanie, 155
- znaków, 141

 class, 276
 

- members, 276



closedir, 172  
 continue, 107  
 cookie, 245, 249  
   odczyt, 250  
   sesji, 263  
   zapamiętywanie danych użytkownika, 250  
   zapisywanie, 246, 250  
   zarządzanie, 249  
   zliczanie liczby odwiedzin, 253  
 Create An Anonymous Account, 362  
 create database, 369  
 CREATE TABLE, 372  
 cudzośćłów, 60  
 czas generowania strony, 138

**D**

dane  
   modyfikacja, 392  
   pobieranie, 384  
   przechowywanie, 57, 353  
   typy, 58  
   usuwanie, 393  
   wprowadzanie, 382  
 data i czas, 132  
   (odwiedziny, 286, 433  
 date, 127  
 DATE, 378, 379  
 DATETIME, 378, 379  
 DCL, 371  
 DDL, 371  
 DECIMAL, 377  
 deklaracja typu dokumentu, 42  
 delDir, 182  
 DELETE, 393  
 destruktory, 284, 295  
 disk\_free\_space, 180  
 disk\_total\_space, 180  
 DML, 371  
 do...while, 103  
 DOCUMENT\_ROOT, 340  
 dodawanie, 69  
 dopisywanie danych na końcu pliku, 197  
 DOUBLE, 377  
 doubleval, 88  
 DROP TABLE, 376  
 dysk  
   serwera (przełglądanie w przeglądarce), 182  
   wolne miejsce, 180  
 dziedziczenie, 288  
 dzielenie, 69  
   modulo, 69

dzień  
   miesiąca, 128, 130, 132  
   tygodnia, 128, 130

**E**

e, 148  
 echo, 44, 50, 64, 144, 228  
 edytor  
   jEdit, 8  
   Notepad++, 8  
   Notepad2, 8  
 elipsa (rysowanie), 332  
 elseif (jedno słowo), 112  
 Enable root access from remote machines, 362  
 Enter the root password, 362  
 ENUM, 381  
 Exception, 304  
 execute, 418  
 explode, 273

**F**

f, 66  
 false, 73  
 fałsz, 73  
 fclose, 188  
 Fedora, 14, 410  
 fetchAll, 416  
 fgetc, 191  
 fgets, 188  
 fields, 276  
 file (formularz), 216  
 file\_exists, 178, 179  
 file\_get\_contents, 194  
 FILE\_IGNORE\_NEW\_LINES, 195  
 file\_put\_contents, 195, 196  
 FILE\_SKIP\_EMPTY\_LINES, 195  
 FILE\_USE\_INCLUDE\_PATH, 195  
 fileatime, 180  
 filectime, 180  
 fileid, 244  
 fileowner, 180  
 fileperms, 180  
 filesize, 179  
 filetype, 180  
 final, 294  
 float, 59, 148  
 FLOAT, 377  
 floatval, 88  
 flock., 203  
 fopen, 178, 186  
   tryby otwarcia pliku, 187  
 for, 100, 108, 109

foreach, 104, 163  
 form feed, 66  
 formatowanie ciągów, 146  
 formularz  
   do wysyłania  
     plików, 224  
     wiadomości, 351  
   elementy  
     składowe, 216  
     typu radio, 219  
   kod, 217  
   odczyt danych  
     GET, 218  
     POST, 222  
   wysyłanie danych metodą POST, 221  
 fpassthru, 194  
 fputs, 195  
 fread, 192, 193  
 fseek, 201  
 ftell, 201  
 FTP, 347  
 ftp\_close, 348  
 ftp\_connect, 347  
 ftp\_get, 349  
 ftp\_login, 347, 349  
 ftpid, 347  
 funkcja, 114  
   argumenty, 115  
   ciało, 114  
   nazwa, 114  
   operująca na ciągach znaków, 389  
   zwracanie wartości, 117  
 fwrite, 195

## G

galeria  
   najprostsza, 316  
   obrazów, 316  
   z miniaturami obrazów, 320  
 GATEWAY\_INTERFACE, 340  
 gd\_info, 316  
 generowaniem odnośników, 242  
 GET, 216  
 getCounter, 206, 207  
 getcwd, 178  
 getdate, 129  
 getimagesize (wartości zwracane), 328  
 głosowanie na ulubione kolory, 428  
 gmdate, 139  
 GMT, 128  
 godzina, 128, 130, 132  
 graficzny licznik odwiedzin, 207  
 GRANT, 370  
 Gutmans Andi, 12

## H

header, 231  
 heredoc, 60, 65  
 hidden (formularz), 216  
 hours, 130  
 HTTP (protokół), 13  
 http://127.0.0.1/, 14, 17  
 http://localhost/, 14, 17  
 HTTP\_ACCEPT, 340  
 HTTP\_ACCEPT\_CHARSET, 340  
 HTTP\_ACCEPT\_ENCODING, 340  
 HTTP\_ACCEPT\_LANGUAGE, 340  
 HTTP\_CONNECTION, 340  
 HTTP\_HOST, 340  
 HTTP\_REFERER, 340  
 HTTP\_USER\_AGENT, 340, 346  
 HTTPS, 340  
 Hypertext Preprocessor, 11  
 HyperText Transfer Protocol, 13

## I

identyfikator  
   sesji, 256  
   strefy czasowej, 128  
 if, 90, 96, 111  
 if...else, 91, 111  
 if...else if, 92, 111  
 iloczyn  
   bitowy, 71  
   logiczny, 74  
 image  
   formularz, 216  
   size  
     x, 328  
     y, 328  
 imagecolorallocate, 326  
 imagecopyresampled, 335  
 imagecopyresized, 335  
 imagecreatefromgif, 325  
 imagecreatefromjpeg, 325  
 imagecreatefrompng, 325  
 imagecreatetruecolor, 325, 330  
 imagedestroy, 325  
 imageellipse, 332  
 imagefilledellipse, 332  
 imagefilledpolygon, 331  
 imagefilledrectangle, 330  
 imagefilter, 334  
 imagegif, 326  
 imagejpeg, 326  
 imageline, 328  
 imagepng, 326

imagepolygon, 331  
 imagerectangle, 330  
 imagerotate, 337  
 imagesx, 328  
 imagesy, 328  
 IMG\_FILTER\_BRIGHTNESS, 334  
 IMG\_FILTER\_COLORIZE, 334  
 IMG\_FILTER\_CONTRAST, 334  
 IMG\_FILTER\_EDGEDETECT, 334  
 IMG\_FILTER\_EMBOSS, 334  
 IMG\_FILTER\_GAUSSIAN\_BLUR, 334  
 IMG\_FILTER\_GRAYSCALE, 334  
 IMG\_FILTER\_MEAN\_REMOVAL, 334  
 IMG\_FILTER\_NEGATE, 334  
 IMG\_FILTER\_PIXELATE, 334  
 IMG\_FILTER\_SELECTIVE\_BLUR, 334  
 IMG\_FILTER\_SMOOTH, 334  
 IN, 388  
 include, 51
 

- lokalizacja dołączanych plików, 56
- różnica w działaniu require, 55

 INDEX, 373  
 index.php, 43  
 informacja
 

- o błędzie, 233
- wyświetlanie, 49

 INSERT, 382  
 instalacja narzędzi
 

- Linux, 14
- Windows, 22

 instanceof, 82  
 instrukcja, 42
 

- if, 90, 111
- if...else if, 92, 111
- if...else, 91, 111
- koniec, 42
- warunkowe zagnieżdżanie, 93
- warunkowa, 90, 111
- wyboru, 97

 integer, 58, 148  
 INTEGER, 377  
 INTO, 382  
 intval, 88, 254  
 InvalidArgumentException, 310, 311  
 IS NOT NULL, 388  
 IS NULL, 388  
 ISO-8859-2, 42

## K

katalog
 

- aktualny, 178
- nawigacja, 183
- odczytanie zawartości, 171

sprawdzenie czy istnieje, 178  
 tworzenie, 176  
 usuwanie, 177  
 wraz z podkatalogami, 182  
 wyświetlanie zawartości, 172  
 zajmowane miejsce, 181  
 zmiana, 177  
 klasa, 276, 279
 

- bazowa, 289
- kontener, 307, 309
- potomna, 288, 290, 295
- składowa, 279

 klucz
 

- główny, 355
- podstawowy, 355

 kod
 

- formatujący, 148
- klasy (automatyczne ładowanie), 284
- źródłowy strony, 43

 kodowanie znaków, 42  
 kolejność wykonywania operacji, 83  
 kolory składowe RGB, 327  
 kolumna (nazwa), 372  
 komentarz, 47
 

- blokowy, 47
- jednowierszowy, 48
- uniksowy, 48

 komórka, 158  
 konfiguracja sesji, 258  
 koniec instrukcji, 42  
 konkatenacja, 80  
 konstruktory, 281, 295
 

- parametry, 283

 kontrast (zmiana poziomu), 335  
 kontrola dostępu (adres IP), 343
 

- do
  - strony, 266
  - witryny, 426
- z wykorzystaniem bazy danych, 426

 kontur prostokąta, 330  
 konwersja, 85
 

- do typu
  - boolean, 88
  - float, 89
  - integer, 89
  - string, 89
- wymuszanie, 87
- zasady, 88

 kończenie połączenia (z bazą danych), 395  
 kropka, 142  
 ksort, 165

**L**

Launch the MySQL Server automatically, 362  
 Leadorf Rasmus, 12  
 lewy ukośnik, 66  
 liczba
 

- dni w miesiącu, 128
- mikrosekund, 128
- minut, 128, 130
- liczba osób aktualnie przeglądających daną stronę, 435
- sekund, 128, 130

 licznik odwiedzin
 

- graficzny, 207
- strony, 423
- tekstowy, 204

 LIKE, 389, 391  
 linia (rysowanie), 328  
 lista
 

- odnośników (plików), 232
- odnośników, 209
- odwiedzin, 210, 433
- plików (automatyczne generowanie), 237
- plików z identyfikatorami, 240

 localtime, 139  
 logiczna
 

- alternatywa wykluczająca, 75
- negacja, 75

 logowanie (procedura), 268  
 ltrim, 151

**Ł**

łańcuch znaków
 

- deklaracja, 60
- przetwarzanie, 152

 łączenie skryptów, 51

**M**

mail, 349, 350  
 mday, 130  
 Mercury Mail Transport System, 350  
 methods, 276  
 metoda
 

- klasa, 276
- finalna, 294
- przesłanie, 293

 miejsce
 

- na dysku, 180
- zajmowane przez katalog, 181

 miesiąc, 128, 130  
 miniatury (obrazów), 320  
 minutes, 130

mkdir, 176  
 mktime, 135  
 mnożenie, 69
 

- przez wielokrotność liczby 2, 73

 modyfikacja danych, 392  
 modyfikator dostępu, 297  
 mon, 130  
 month, 130  
 MySQL, 8, 353
 

- Client programs and shared libraries, 364
- instalacja, 360
  - Linux, 363
  - Windows, 360
- operatory
  - logiczne, 388
  - relacyjne, 387

 mysql\_affected\_rows, 400  
 mysql\_close, 395  
 mysql\_connect, 394, 396, 401  
 mysql\_fetch\_array, 397, 398, 400  
 mysql\_fetch\_row, 397, 398  
 mysql\_num\_rows, 397  
 mysql\_query, 397  
 mysql\_select\_db, 396  
 mysqladmin, 364  
 mysqld, 366

**N**

n, 66  
 nagłówek, 42  
 napis (wyświetlenie), 41  
 nawiązanie połączenia z bazą danych, 394  
 nawigacja po katalogach, 183  
 nazwa
 

- dnia tygodnia, 132
- funkcji, 114
- miesiąca, 128, , 132
- zmiennych iteracyjnych, 109

 negacja bitowa, 72  
 new, 83, 279
 

- line, 66

 New Root password, 362  
 newdoc, 61  
 nl2br, 146  
 NOT, 72, 75, 389
 

- BETWEEN N AND M, 388
- IN, 388
- LIKE, 389
- NULL, 373, 383

 notacja wykładnicza, 148  
 nowa linia, 66  
 null, 381  
 NULL, 383

- numer
  - dnia w roku, 129, 130
  - tygodnia w roku, 128
  
- O**
  
- obiekt, 275, 279
  - niszczenie, 284
  - usuwanie z pamięci, 284
- obracanie (obraz), 337
- obraz, 334
  - kolory, 326
  - obracanie, 337
  - rozmiar, 328
  - skalowanie, 335
  - szerokość, 328
  - tworzenie, 325
  - usuwanie, 325
  - wysokość, 328
  - zapisywanie do pliku, 326
- odczyt
  - bajtów z pliku binarnego, 192
  - danych z pliku binarnego, 192
  - pełnej zawartości pliku, 193
  - z pliku pojedynczego
    - wiersza, 188
    - znaku, 191
  - zawartości katalogu, 171
- oddawanie głosów, 428
- odejmowanie, 69
- odnośnik
  - generowanie, 242
  - lista, 209
- odwiedziny (zbieranie informacji), 286
- opendir, 171
- operacja
  - na bitach, 70
  - przypisania, 68
- operator
  - arytmetyczny, 69
  - bitowy, 70
  - dekrementacji, 76
  - indeksowania tablicy, 80
  - inkrementacji, 76
  - kontroli błędów, 81
  - kontroli typów, 82
  - logiczny, 73
    - w MySQL, 388
  - łańcuchowy, 80
  - łączenia
    - ciągów, 142
    - tablic, 79
  - porównywania, 76
    - tablic, 80
  - przypisania, 75
  - relacyjny, 76
    - w MySQL, 387
  - rozdzielania wyrażień, 83
  - rzutowania typów, 82
  - tablicowy, 79
  - tworzenia obiektów, 83
  - warunkowy, 81, 99
    - wykonania polecenia zewnętrznego, 81
- OR, 71, 74, 388
- ORDER BY, 385
- ORG\_PATH\_INFO, 341
- otwieranie pliku, 186
  
- P**
  
- password (formularz), 216
- PATH\_INFO, 341
- PATH\_TRANSLATED, 341
- PDO, 410, 411
  - kończenie połączenia z bazą, 412
  - nawiązywanie połączenia z bazą, 411
  - pobranie pełnych wyników zapytania, 416
  - zapytanie
    - modyfikujące dane, 418
    - pobierające dane, 412
- Personal HomePage Tools, 11
- pętla, 100
  - do...while, 103
  - for, 100, 108, 112
  - foreach, 104, 112
  - while, 102, 112
  - zagnieżdżona, 109
- PHP, 8, 11
  - historia, 12
  - instalacja
    - Linux
      - Fedora, 15
      - Ubuntu, 20
    - Windows, 24
  - plik konfiguracyjny
    - Linux, 38
    - Windows, 38
    - XAMPP, 38
  - testowanie
    - Linux
      - Fedora, 18
      - Ubuntu, 21
    - Windows, 27

## PHP

- wiersz poleceń
    - Linux, 39
    - Windows, 39
    - XAMPP, 31
  - PHP Data Objects, 410
  - PHP\_AUTH\_DIGEST, 341
  - PHP\_AUTH\_PW, 341
  - PHP\_AUTH\_USER, 341
  - PHP\_SELF, 339
  - php5-mysql, 410
  - php5-pdo, 410
  - php-pdo, 410
  - plik
    - binarny odczyt danych, 192
    - dopisywanie danych na końcu, 197
    - informacje, 180
    - odczyt pełnej zawartości, 193
    - otwieranie, 186
    - pobranie z serwera, 347
    - prawa dostępu, 180
    - rozmiar, 179
    - sprawdzenie czy istnieje, 179
    - synchronizacja dostępu, 202
    - tworzenie, 178, 186
    - typ, 180
    - usuwanie, 178
    - właściciel, 180
    - zamykanie, 188
    - zapis danych, 195
  - plus, 142
  - pm, 127
  - PM, 127
  - pobieranie
    - danych, 384
    - plików
      - w serwisie WWW, 232
      - z serwera, 347, 348
    - wszystkich wierszy tabeli, 385
    - zawartości wybranych kolumn, 386
  - poczta (wysyłanie), 349
  - podzielenie przez wielokrotność liczby 2, 73
  - pojedynczy
    - wiersz (odczyt z pliku), 188
    - znak (odczyt z pliku), 191
  - pole (klasa), 276
  - polskie nazwy miesięcy, 131
  - połączenie
    - z serwerem, 401
    - zamykanie (FTP), 348
  - położenie wskaźnika pozycji w pliku, 201
  - porównanie ciągów, 154
  - POST, 221
  - PostCast Serve, 350
  - powrót karetki, 66
  - prawa dostępu do pliku, 180
  - prawda, 73
  - primary key, 355, 373
  - primitive types, 58
  - print, 50, 228
  - printf, 148, 228
  - printList, 242
  - priorytety operatorów, 83
  - private, 297
  - procedura
    - logowania, 268
    - wylogowania, 271
  - properties, 276
  - prostokąt
    - kontur, 330
    - rysowanie, 330
    - wypełniony kolorem, 330
  - protected, 297
  - przechowywanie danych, 57, 353
  - przechwytywanie
    - wielu wyjątków, 311
    - wyjątków (kolejność), 312
  - przeglądanie w przeglądarce zawartości dysku
    - serwera, 182
  - przeglądarka
    - odwiedziny, 286
    - rozpoznanie typu, 345
  - przekątna (rysowanie), 330
  - przesłanie metod, 293
  - przesunięcie
    - bitowe w
      - lewo, 73
      - prawo, 73
    - strefy czasowej, 129
  - przeszukiwanie ciągów, 155
  - przetwarzanie danych, 359
  - public, 297
- Q**
- query, 420
  - QUERY\_STRING, 340
  - queryExec, 422
  - QuickSort (algorytm sortowania), 168
- R**
- r, 66, 187
  - r+, 187
  - radio (formularz), 216
  - RDBMS, 353
  - readData, 435
  - readdir, 172

readfile, 194  
 relacja, 355  
   jeden do jednego, 356  
   wiele do wielu, 357  
 REMOTE\_ADDR, 341  
 REMOTE\_HOST, 341  
 REMOTE\_PORT, 341  
 REQUEST\_METHOD, 340  
 REQUEST\_TIME, 340  
 REQUEST\_URL, 341  
 require, 51, 54  
   lokalizacja dołączanych plików, 56  
   różnica w działaniu include, 55  
 reset (formularz), 216  
 return, 117  
 Retype the password (Confirm), 362  
 rewind, 201, 202  
 rmdir, 177  
 rodzaj przeglądarki, 433  
 rok, 128, 129, 130  
   czterocyfrowy, 132  
   przestępny, 128  
 rozmiar pliku, 179  
 rozpoznanie typu przeglądarki internetowej, 345  
 różnica symetryczna, 72, 75  
 rtrim, 151  
 rysowanie  
   elipsa, 332  
   linia, 328  
   prostokąt, 330  
   przekątna, 330  
   wielokąt, 331  
   wycinki elips, 332

## S

scalar types, 58  
 scandir, 174  
 schemat przetwarzania danych, 359  
 SCRIPT\_FILENAME, 341  
 SCRIPT\_NAME, 341  
 seconds, 130  
 SELECT, 384, 386, 387  
 select (formularz), 216  
 selektywne pobieranie danych, 387  
 send, 244  
 Sendmail, 349  
 SENDMAIL\_FROM, 350  
 sendmail\_path, 349  
 SERVER\_ADDR, 340  
 SERVER\_ADMIN, 341  
 SERVER\_NAME, 340  
 SERVER\_PORT, 341  
 SERVER\_PROTOCOL, 340

SERVER\_SIGNATURE, 341  
 SERVER\_SOFTWARE, 340  
 serwer, 14  
   MySQL  
     komunikacja, 368  
     konto użytkownika (tworzenie), 370  
     uruchamianie  
       Linux, 366  
       Windows, 365  
       XAMPP, 367  
     zarządzanie bazą danych, 369  
       Linux, 367  
       Windows, 367  
       XAMPP, 368  
 WWW  
   brak połączenia, 28  
   instalacja  
     Linux  
       Fedora, 14  
       Ubuntu, 19  
     Windows, 22  
   nie działa po instalacji PHP, 28  
   nie obsługuje PHP, 29  
   problemy z uruchamianiem, 28  
   restartowanie  
     Linux, 21  
     Windows, 25  
   testowanie  
     Linux  
       Fedora, 17  
       Ubuntu, 21  
       Windows, 27  
   uruchamianie  
     Linux  
       Fedora, 17  
       Ubuntu, 21  
     Windows, 25  
   XAMPP, 31  
   zatrzymywanie  
     Linux, 21  
     Windows, 25  
 sesja, 256  
   cookie, 263  
   identyfikator, 256  
   implementacja, 260  
   konfiguracja, 258  
   kontrola dostępu do strony, 266  
   kończenie, 257  
   rozpoczynanie, 256  
   śledzenie zachowań użytkownika, 264  
   zakończenie, 263  
   zmienne, 257

- session.auto\_start, 258
- session.cache\_expire, 258
- session.cache\_limiter, 258
- session.cookie\_domain, 258
- session.cookie\_httponly, 258
- session.cookie\_lifetime, 258
- session.cookie\_path, 258
- session.cookie\_secure, 258
- session.entropy\_file, 258
- session.entropy\_length, 258
- session.gc\_divisor, 259
- session.gc\_maxlifetime, 259
- session.gc\_probability, 259
- session.hash\_bits\_per\_character, 259
- session.hash\_function, 259
- session.name, 259
- session.save\_handler, 259
- session.save\_path, 260
- session.serialize\_handler, 260
- session.use\_cookies, 260
- session.use\_only\_cookies, 260
- session.use\_trans\_sid, 260
- session\_destroy, 257, 263
- session\_get\_cookie\_params, 263
- session\_start, 256, 261
- SET, 381
- setcookie, 252
- Set-Cookie, 245
- skalowanie
  - obraz, 335
  - schemat, 336
- składnia alternatywna, 111
- składowe, 292
  - klasy, 276
  - statyczne, 299
    - odwoływania na zewnątrz klasy
      - przez istniejący obiekt, 301
      - sposób odwoływania, 300
- skrypt
  - odbierający pliki, 226
  - PHP
    - oddzielenie od kodu HTML, 45
    - zadanie, 44
- skrypty (łączenie), 51
- SMTP, 350
- SMTP\_PORT, 350
- sort, 164
- sortowanie, 164
  - tablica asocjacyjna, 165
  - wyników, 385
- specyfikator dostępu, 297
- sposób kodowania znaków, 42
- sprawdzenie czy istnieje
  - katalog, 178
  - plik, 179
- SQL, 371
- SQLite, 8, 354, 402
  - funkcja sqlite\_array\_query, 408
  - instalacja, 359
  - kończenie połączenia, 402
    - z bazą, 420
  - nawiązania połączenia, 402
    - z bazą, 419
  - wykonywanie zapytań, 403
  - zapytania
    - modyfikujące dane, 406, 422
    - pobierająca dane, 404, 420
- sqlite\_, 404
- sqlite\_array\_query, 408
- sqlite\_close, 402
- sqlite\_exec, 406
- sqlite\_fetch\_all, 404
- sqlite\_fetch\_array, 404, 406
- sqlite\_open, 402
- sqlite\_query, 403
- sqlite\_unbuffered\_query, 403
- SQLiteDatabase, 419
- start, 155
- static, 299
- statyczne składowe, 299
- str\_ireplace, 152, 153
- str\_replace, 152
- strcasecmp, 155
- strcmp, 154
- strefa czasowa, 128
- strftime, 132
- string, 59, 148
- stripos, 155
- stristr, 155
- strona (kod źródłowy), 43
- strpos, 155
- stripos, 155
- strrpos, 155
- strstr, 155
- strtolower, 150
- strtotime, 137
- strtoupper, 150
- strtr, 152, 153
- Structured Query Language, 371
- struktura tabeli (baza danych), 354
- strval, 88
- submit (formularz), 216
- substr\_replace, 152, 153
- suma bitowa, 71
- suma logiczna, 74
- superglobal, 119
- Suraski Zeev, 12
- Swatch, 127
- switch, 97



switch...case, 97  
 sygnalizacja wystąpienia błędu, 305  
 synchronizacja dostępu do plików, 202  
 system  
   kontrola dostępu do witryny, 426  
   operacyjny (rozpoznawanie), 346  
 sześciokąt, 331

## Ś

śledzenie  
   liczby odwołań do podstron witryny, 265  
   zachowań użytkownika, 264

## T

t, 66  
 tabela  
   modyfikacja struktury, 374  
   nazwa, 372  
   tworzenie, 372  
   usuwanie, 376  
 tabela (baza danych), 354  
   powiązania, 355  
 tablica, 158  
   \$\_SERVER, 339  
   asocjacyjna, 161  
     odczyt, 162  
     sortowanie, 165  
   liczba elementów, 164  
   odczyt zawartości, 159  
   sortowanie, 164  
   wyświetlanie zawartości, 163  
   zmiana wartości komórek, 160  
 tabulator  
   pionowy, 66  
   poziomy, 66  
 tekstowy licznik odwiedzin, 204  
 TEXT, 381  
 text (formularz), 216  
 textarea (formularz), 216  
 The MySQL server and related files, 364  
 throw, 303, 304  
 thumbnails., 320  
 time, 140  
 TIME, 379  
 TIMESTAM, 379  
 timestamp, 127, 137  
 TIMESTAMP, 378  
 tm\_hour, 140  
 tm\_isdst, 140  
 tm\_mday, 140  
 tm\_min, 140  
 tm\_mon, 140

tm\_sec, 140  
 tm\_wday, 140  
 tm\_yday, 140  
 tm\_year, 140  
 treść, 42  
 treść skryptu PHP, 42  
 trim, 151  
 true, 73  
 try, 306  
 try...catch, 310  
 tryby otwarcia pliku, 187  
 tworzenie  
   dynamicznych stron internetowych, 7  
   katalogu, 176  
   pliku, 178, 186  
 typ  
   całkowitoliczbowy, 58  
   danych, 377  
   daty i czasu, 378  
   float, 59  
   liczbowy, 377  
   logiczny, 58  
   łańcuchowy, 379  
   null, 62  
   pliku, 180  
   prosty, 58  
   resource, 62  
   skalarny, 58  
   specjalny, 62  
   string, 59  
   złożony, 62  
 tytuł strony, 42

## U

uasort, 167  
 Ubuntu, 19, 410  
 ucfirst, 150  
 ucwords, 150  
 uksort, 167  
 unbuffered\_query, 420  
 UNIQUE, 373  
 unlink, 178  
 unset, 262  
 unset(\$db);, 420  
 UNSIGNED, 378  
 UPDATE, 392  
 use, 371  
 usort, 167, 168  
 usuwanie  
   danych, 393  
   katalogu, 177  
     wraz z podkatalogami, 182  
   plików, 178

UTF-8, 42  
 użytkownik  
   śledzenie zachowań, 264  
   weryfikacja, 266

**V**

v, 66  
 VARBINARY, 380  
 VARCHAR, 380  
 vprintf, 228

**W**

w, 187  
 w+, 187  
 wartość  
   pusta, 381  
   wyświetlana (oddzielanie przecinkami), 50  
 warunkowe wyrażenie, 96  
 wday, 130  
 weekday, 130  
 wektor, 158  
 weryfikacja użytkownika (nazwa, hasło), 266  
 WHERE, 392  
 while, 102  
 widoczność zmiennych, 118  
 wiek, 132  
 wielkość liter, 150  
 wielokąt (rysowanie), 331  
 własne wyjątki, 313  
 właściciel pliku, 180  
 właściwość (klasa), 276  
 wolne miejsce na dysku, 180  
 wprowadzanie danych, 382  
 writeData, 212  
 wskaźnik pozycji w pliku, 201  
 wstawienie treści wskazanego pliku, 51  
 wybór bazy, 396  
 wycinek elipsy (rysowanie), 332  
 wyjątek, 303, 304  
   przechwytywanie, 305, 310, 311  
   własny, 313  
   wyrzucanie, 303  
   zgłoszenie, 305  
 wykonywanie  
   operacji na bitach, 70  
   zapytań, 397  
 wylogowanie (procedura), 271  
 wyniki (sortowanie), 385  
 wyrażenie warunkowe, 96  
 wyrzucanie wyjątku, 303

wysłanie pliku  
   do przeglądarki, 230  
   z komputera użytkownika na serwer, 223  
 wysunięcie strony, 66  
 wysyłanie  
   danych do przeglądarki, 228  
   plików do  
     przeglądarki, 228  
     użytkownika, 234  
   poczty, 349  
   wiadomości, 351  
 wyświetlanie  
   czasu, 127  
   daty, 127  
   informacji, 49  
   napisu, 41  
   zawartości katalogu, 172

**X**

x, 187  
 x+, 187  
 XAMPP, 8, 31, 354, 365  
   bezpieczeństwo, 33  
     (hasła dostępu), 36  
     (nazwa użytkownika), 38  
   instalacja  
     Linux, 31  
     Windows, 34  
   restartowanie serwerów (Linux), 32  
   testowanie  
     Linux, 32  
     Windows, 36  
   uruchomienie serwerów (Linux), 32  
   zarządzanie (Windows), 34  
   zatrzymanie serwerów (Linux), 32  
 XHTML 1.0 Strict, 42  
 XOR, 72, 75, 389

**Y**

yday, 130  
 year, 130  
 YEAR, 379

**Z**

zadanie skryptu PHP, 44  
 zagnieżdżanie  
   instrukcji warunkowych, 93  
   pętli, 109  
 zakończenie sesji, 263

- zamykanie
  - pliku, 188
  - połączenie (ftp), 348
- zapamiętywanie danych użytkownika, 250
- zapis do pliku, 195
- zapytania, 397
  - aktualizujące dane, 400
  - pobierające dane, 397
- zasięg zmiennych, 118
- Zend, 12
- ZEROFILL, 378
- zintegrowane środowisko programistyczne
  - EasyEclipse for PHP, 8
  - NetBeans, 8
- zliczania liczby odwiedzin, 253
- złączenie, 80
- złożone wyrażenia warunkowe, 96
- zmiana
  - katalogu bieżącego, 177
  - nazw kolumn w wynikach zapytania, 386
  - poziomu kontrastu, 335
- zmienna, 57
  - autoglobalna, 83, 123
  - globalna, 83, 118
  - iteracyjna (nazwy), 109
  - lokalna, 121
  - nazwa, 57
  - operacje, 68
  - sesji, 257
  - statyczna, 121
  - superglobalna, 83, 119, 123
  - typ, 58
  - w kodzie skryptu, 62
  - widoczność, 118
  - wyświetlanie wartości, 64
  - zasięg, 118
  - zmiennoprzecinkowy, 377
- znacznik
  - czasu, 127, 135, 180
    - Uniksa, 128, 130
  - kanoniczny, 45
  - kończący akapit, 44
  - otwierający, 45
  - skryptów HTML, 46
  - typu
    - ASP, 46
    - SGML, 46
  - w postaci skróconej, 46
  - zamykający, 45
- znak
  - 0, 151
  - cudzysłowu, 66
  - dolara, 66
  - nowej linii, 151
  - powrotu karetki, 151
  - spacji, 151
  - tabulacji, 151
- zwracanie wartości, 117

Znajomość języka skryptowego PHP, szczególnie w wersji PHP 5, to w dzisiejszych czasach standard, bez którego nie może się obyć żaden szanujący się twórca stron internetowych. Środowisko PHP 5 pozwala tworzyć dynamiczne witryny, efektywnie komunikujące się z bazami danych. Zapewnia też możliwość współpracy Twojej strony z różnymi rodzajami danych, a ponadto nadaje się do tworzenia samodzielnie działających aplikacji. Jeśli marzy Ci się kariera webmastera, a nie masz jeszcze odpowiednich umiejętności w tej dziedzinie lub chcesz odświeżyć wiedzę, ta książka umożliwi Ci szybkie wejście na grunt praktycznego zastosowania możliwości języka PHP w Twoich własnych projektach.

W podręczniku „PHP 5. Praktyczny kurs. Wydanie II” znajdziesz kompletne informacje o podstawach PHP — od kwestii związanych z nazewnictwem, instalacją i konfiguracją niezbędnych narzędzi, przez omówienie zasad budowy skryptów, aż po programowanie obiektowe i obsługę sieci. Dowiesz się, jak uruchomić działającą (i atrakcyjną) witrynę internetową, jak wykorzystywać możliwości grafiki i o czym należy pamiętać, aby uniknąć kłopotów z wyświetlaniem witryny w przeglądarce. Nauczysz się obsługiwać protokoły sieciowe i zrozumiesz, na czym polega obsługa sesji. Sprawdź, jak wiele możesz osiągnąć, programując w PHP!

- Instalacja i konfiguracja narzędzi
- Pierwszy skrypt
- Zmienne, typy danych i operatory
- Instrukcje sterujące i funkcje
- Obsługa daty i czasu, ciągi znaków, tablice
- Operacje na strukturze systemu plików
- Operacje na plikach i praktyczne wykorzystanie plików
- Odbieranie danych z przeglądarki
- Wysyłanie danych do przeglądarki
- Obsługa cookies, sesje i wyjątki
- Programowanie obiektowe
- Obsługa grafiki
- Połączenia sieciowe, poczta i FTP
- PHP i popularne bazy danych
- Obiektowa współpraca z MySQL i SQLite
- Bazy danych w praktyce

**PHP 5 — po prostu musisz to znać!**

**helion.pl**  
księgarnia internetowa

Nr katalogowy: **6604**



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/nnowosci>

**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>

sięgnij po **WIĘCEJ**



**KOD KORZYŚCI**

ISBN 978-83-246-3393-7



**Cena 69,00 zł**

**Informatyka w najlepszym wydaniu**