PHP4. Leksykon kieszonkowy

Autor: Rasmus Lerdorf
Tłumaczenie: Daniel Kaczmarek
ISBN: 83-7361-074-X
Tytuł oryginału: PHP Pocket Reference
Format: B5, stron: 192

PHP jest popularnym językiem skryptowym, dostępnym na wielu platformach na licencji Open Source. PHP można bezpośrednio osadzać w kodzie HTML, tworząc złożone aplikacje WWW, korzystające w prosty sposób z rozmaitych baz danych.

Jedną z charakterystycznych cech PHP jest ogromna liczba funkcji dostępnych w tym języku. Dzięki książce „PHP4. Leksykon kieszonkowy" będziesz miał ich opisy zawsze pod ręką. W książce tej znajdziesz również opisy struktur kontrolnych, zmiennych, typów i operatorów PHP, a także kilka prostych, praktycznych przykładów ilustrujących rozwiązania najczęściej spotykanych problemów.

Rasmus Lendorf stworzył język PHP w roku 1995 i od tego czasu aktywnie uczestniczy w jego rozwoju. Ma ponad dziesięcioletnie doświadczenie w programowaniu, a obecnie zatrudniony jest w firmie IBM na stanowisku starszego inżyniera oprogramowania.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Typy danych

PHP udostępnia cztery podstawowe typy danych: całkowitoliczbowy, liczb zmiennopozycyjnych, łańcuchów znaków oraz logiczny. Ponadto dostępne są dwa złożone typy danych: tablice oraz obiekty.

Całkowitoliczbowy

Jest to typ liczb niezawierających części ułamkowej. Zakres liczb całkowitych w PHP jest taki sam jak zakres typu danych `long` w języku C. Na platformie 32-bitowej liczby całkowite należą do zakresu od –2 147 483 648 do +2 147 483 647. Jeśli przypadkiem przekroczony zostanie ten zakres liczbowy, PHP automatycznie przekształci taką liczbę do liczby zmiennopozycyjnej. Liczba całkowita może zostać zapisana w systemie dziesiętnym (o bazie 10), szesnastkowym (o bazie 16) lub ósemkowym (o bazie 8), na przykład:

```
$decimal=16;
$hex=0x10;
$octal=020;
```

Liczby zmiennopozycyjne

Liczby zmiennopozycyjne posiadają część ułamkową. Zakres liczb zmiennopozycyjnych w PHP jest taki sam jak zakres typu danych `double` w języku C. Na większości platform liczba typu `double` należy do przedziału od 1,7E-308 do 1,7E+308. Liczba taka może być wyrażana w postaci zwykłej liczby z częścią ułamkową bądź też przy użyciu notacji naukowej, na przykład:

```
$var=0.017;
$var=17.0E-3;
```

PHP posiada również dwa zbiory funkcji służących do wykonywania operacji na liczbach o precyzji z góry określonej. Zbiory te noszą nazwę funkcji BC oraz funkcji GMP. Więcej informacji na ten temat znajdziesz pod adresem *http://www.php.net/bc* oraz *http://www.php.net/gmp*.

Łańcuchy znaków

Łańcuch znaków jest sekwencją znaków. Łańcuch znaków może być ograniczony cudzysłowem pojedynczym lub cudzysłowem podwójnym:

```
'PHP jest cool'
"Witaj świecie!"
```

W przeciwieństwie do łańcuchów znaków ograniczonych pojedynczymi cudzysłowami, łańcuchy znaków ograniczone podwójnymi cudzysłowami mogą być poddawane zastępowaniu przez zmienne, możliwe jest również obsługiwanie w nich sekwencji znaków unikowych, na przykład:

```
$a="świecie";
echo "Witaj\t$a\n";
```

Powyższy kod spowoduje wyświetlenie słowa "Witaj", po którym występować będzie znak tabulacji, a następnie słowo "świecie" oraz znak nowej linii. Innymi słowy zastępowanie przez zmienną jest wykonywane na zmiennej $a, a sekwencje znaków są przekształcane na odpowiadające im znaki. Inaczej będzie w poniższym przypadku:

```
echo 'Witaj\t$a\n';
```

W takiej sytuacji wyświetlone zostanie "Witaj\t$a\n". Zastępowanie przez zmienną nie ma miejsca, a sekwencje znaków unikowych nie są obsługiwane.

Kolejnym sposobem przypisywania łańcucha znaków jest zastosowanie tak zwanej składni *heredoc*. Zaletą takiego podejścia jest fakt, iż nie istnieje konieczność poprzedzania cudzysłowów znakami unikowymi. Można to przedstawić następująco:

```
$foo = <<<EOD
  To jest łańcuch znaków "multiline" mający kilka
  ↳linii
  przypisany z zastosowaniem składni 'heredoc'.
EOD;
```

Poniższa tabela przedstawia sekwencje unikowe obsługiwane przez PHP wewnątrz łańcuchów znaków ograniczanych podwójnymi cudzysłowami.

Sekwencja unikowa	Znaczenie
\n	Nowa linia (LF lub 0x0A (10) w kodzie ASCII)
\r	Powrót karetki (CR lub 0x0D (13) w kodzie ASCII)
\t	Tabulator poziomy (HT lub 0x09 (9) w kodzie ASCII)
\\	Lewy ukośnik
\$	Znak dolara
\"	Podwójny cudzysłów
\123	Reprezentacja znaku w notacji ósemkowej
\x12	Reprezentacja znaku w notacji szesnastkowej

Typ logiczny

Typ logiczny ma tylko dwie wartości: `true` i `false`. Oto przykład:

```
$flag = true;
```

Wartości logiczne są używane najczęściej, wówczas gdy wykonywane jest porównanie przy użyciu operatorów == lub === oraz zwracany jest jego wynik.

Tablice

Tablica jest złożonym typem danych, mogącym zawierać wielokrotne wartości danych indeksowanych liczbowo lub za pomocą łańcuchów znaków. Tablica łańcuchów znaków może być zapisana na przykład w następujący sposób:

```
$var[0]="Witaj";
$var[1]="świecie";
```

Zwróć uwagę, że przypisując elementy tablicy przy użyciu tej metody, nie musisz stosować do ich indeksowania kolejnych liczb.

PHP pozwala również na dodawanie elementu na końcu tablicy „na skróty", czyli bez wskazywania indeksu, na przykład:

```
$var[]="Test";
```

PHP nada temu elementowi następny logiczny indeks liczbowy. W tej sytuacji elementowi `"Test"` w tablicy `$var` zostanie nadany indeks 2: jeśli w tablicy występują elementy niebędące elementami kolejnymi, PHP nada wartość indeksu o jeden większą niż aktualna najwyższa wartość indeksu. Opisany mechanizm automatycznego indeksowania jest najbardziej użyteczny, wówczas gdy ma się do czynienia z elementami wielokrotnego wyboru `<select>` formy, o czym przekonamy się za jakiś czas na przykładzie.

Łańcuchy znaków określiliśmy, co prawda, jako podstawowy typ danych, jednak możliwe jest również potraktowanie łańcucha znaków jako złożonego typu danych, w którym dostęp można uzyskiwać osobno do każdego znaku łańcucha. Inaczej mówiąc, łańcuch znaków można traktować jak tablicę znaków, w której pierwszy znak ma indeks o numerze 0. Z łańcucha można więc wyodrębnić trzeci znak:

```
$string[2]
```

W celu uniknięcia niejednoznaczności między łańcuchami znaków oraz tablicami wprowadzono nową składnię odwoływania się do pojedynczych znaków łańcuchów:

```
$string{2}
```

Składnia ta odpowiada zapisowi `$string[2]` i jest zalecana.

Tablice mogą być także indeksowane przy użyciu łańcuchów znaków. Tablice takie nazywane są *tablicami asocjacyjnymi*:

```
$var["Styczeń"]=1;
$var["Luty"]=2;
```

W jednej tablicy można stosować indeksy zarówno liczbowe, jak i w postaci łańcuchów znaków, ponieważ wewnątrz PHP wszystkie tablice są traktowane jak tablice asocjacyjne, w których indeksy mogą mieć dowolną postać.

Przez każdą tablicę w PHP można bezpiecznie przechodzić, stosując następujący mechanizm:

```
foreach($array as $key=>$value) {
  echo "array[$key]=$value<br>\n";
  }
```

Jest to najczęściej wykorzystywany sposób odczytywania każdego elementu tablicy bez względu na to, czy jest ona tablicą indeksowaną numerycznie, czy też tablicą asocjacyjną. PHP zawiera szereg funkcji służących do operowania na tablicach — zostaną one szczegółowo przedstawione w „Leksykonie funkcji".

Obiekty

Obiekt jest złożonym typem danych, który może zawierać dowolną liczbę zmiennych oraz funkcji. Obsługa obiektów w wersji 4. języka PHP jest dosyć ograniczona. Wersja 5. języka usprawnia mechanizmy charakterystyczne dla orientacji obiektowej. W PHP4 obsługa orientacji obiektowej została tak zaprojektowana, by ułatwić kapsułkowanie struktur danych oraz funkcji w celu zamknięcia ich w klasach możliwych do wielokrotnego wykorzystywania. Oto prosty przykład:

```
class test {
  var $str = "Witaj świecie";
  function init($str) {
    $this->str = $str;
  }
}

$class = new test;
echo $class->str;
$class->init("Witaj");
echo $class->str;
```

Powyższy kod tworzy obiekt `test` przy użyciu operatora `new`. Następnie w ramach tego obiektu ustawiana jest zmienna o nazwie `str`. Używając terminologii obiektowej, można powiedzieć, iż zmienna wewnątrz obiektu jest właściwością tego obiektu. Obiekt `test` definiuje również funkcję, czyli metodę o nazwie `init()`. Metoda ta stosuje specjalną zmienną `$this` w celu dokonania zmiany wartości właściwości `str` tego obiektu.

Dziedziczenie jest obsługiwane przy użyciu słowa kluczowego `extends` w definicji klasy. Możemy rozszerzyć przedstawioną wcześniej klasę `test` w następujący sposób:

```
class more extends test {
  function more() {
    echo "Wywołano konstruktor";
  }
}
```

Oznacza to, iż klasa `more` dziedziczy po klasie `test`, a ponadto wprowadza pojęcie konstruktora. Jeśli znajdująca się wewnątrz klasy metoda nosi tę samą nazwę co klasa, staje się ona funkcją konstruktora dla tej klasy. Konstruktor jest wywoływany automatycznie w momencie tworzenia egzemplarza klasy.

Znacznie więcej informacji można znaleźć pod adresem *http://www.php.net/oop*.

Rzutowanie typu

Jak już wcześniej wspominaliśmy, nie musisz określać typu w momencie tworzenia zmiennej, jednak nie oznacza to wcale, że zmienne nie posiadają przypisanych im typów. Możesz otwarcie określić typ, czyli wykonać rzutowanie typu, stosując składnię znaną z języka C, w której nazwę pożądanego typu umieszcza się w nawiasach przed zmienną lub wyrażeniem, na przykład:

```
$var = (int)"123abc";
```

Gdyby w powyższym kodzie zabrakło fragmentu `(int)`, PHP utworzyłby zmienną będącą łańcuchem znaków. Jednak ze względu na otwarte wskazanie typu utworzyliśmy zmienną typu całkowitoliczbowego o wartości 123. Poniższa tabela przedstawia operatory rzutowania dostępne w PHP.

Operatory	Funkcja
`(int), (integer)`	Rzutowanie do typu całkowitoliczbowego
`(real), (double), (float)`	Rzutowanie do typu liczby zmiennopozycyjnej
`(string)`	Rzutowanie do typu łańcucha znaków
`(array)`	Rzutowanie do typu tablicowego
`(object)`	Rzutowanie do obiektu
`(bool), (boolean)`	Rzutowanie do typu logicznego
`(unset)`	Rzutowanie do wartości `NULL`; odpowiada wywołaniu funkcji `unset()` względem wartości

Chociaż nie są zazwyczaj niezbędne, PHP udostępnia jednak następujące wbudowane funkcje służące do sprawdzania w kodzie programu typów zmiennych: `gettype()`, `is_bool()`, `is_long()`, `is_float()`, `is_string()`, `is_array()` oraz `is_object()`.

Wyrażenia

Wyrażenia są podstawowym środkiem konstrukcyjnym w ramach języka. Wszystko, co posiada wartość, może być traktowane jako wyrażenie. Poniżej przytaczamy następujące przykłady:

```
5
5+5
$a
$a==5
sqrt(9)
```

Łącząc większą liczbę takich podstawowych wyrażeń, można tworzyć wyrażenia większe i bardziej złożone.

Zwróć uwagę, iż instrukcja `echo` stosowana przez nas w licznych przykładach nie może być częścią wyrażenia złożonego, ponieważ nie zwraca żadnej wartości. Z drugiej strony instrukcja `print` może być zastosowana jako część wyrażenia złożonego, ponieważ zwraca wartość. Pod każdym innym względem instrukcje `echo` oraz `print` są identyczne: przekazują dane wyjściowe.