Лекція №8 (частина 1)

"*Типи даних мови PHP*

Викл. Коваль В.В.

ОСІТ

2021р.

Синтаксис мови PHP. Типи даних. Керуючі послідовності. Оператори.

Типи даних

Скалярні типи:

- **boolean (логічний);**
- **integer (цілий);**
- **float (з плаваючою точкою);**
- **string (рядковий).**

Змішані типи:

- **array (массив);**
- **object (об'єкт);**
- callable;
- iterable;

Спеціальні типи:

- **resource (ресурс);**
- **NULL;**

Псевдотипи:

- mixed (змішаний);
- number (числовий);
- callback (зворотнього виклику);
- array|object;
- void;

Синтатксис

```
<?php
echo "Hello, world!"; // коментар
/* Багаторядковий
коментар*/
echo "Я вивчаю PHP ";
  # Це коментар у стилі
  # Unix
?>
```

Змінні

```
$first = ' Text ';
$second = &$first;
```

Константи

```
define(“Ім`я_константи”,      "Значення_константи",
    [Нечутливість_до_регистру])

define("GREETING", "Hello you.", true);
    echo GREETING; //  "Hello you."
    echo Greeting; //    "Hello you."
```

Суперглобальний масив $_REQUEST

```
<?php
$str = "Добрий день,"
.$_REQUEST["first_name"]. "  "
.$_REQUEST["last_name"]."! <br>";
$str .=" Ви вибрали для вивчення курс по    ".$_REQUEST["kurs"];
echo $str;
?>
```

Масиви $HTTP_POST_VARS, $HTTP_GET_VARS, $_POST, $_GET

```
<?php
$str = "Добрий день,    ".$_POST ["first_name"]."    ".$_POST ["last_name"] ."!
    <br>";
$str .= "Ви вибрали для вивчення курс по ".       $_POST["kurs"];
echo $str;
?>
```

Функція getenv()

```
<?
getenv('REQUEST_METHOD');    // повертає метод
echo getenv ('REMOTE_ADDR');    // виводить IP-адресу користувача
?>
```

```php
<?php
    define("PASSWORD","qwerty");
    define("PI","3.14", true);
    echo (PASSWORD);
    echo constant("PASSWORD");
    echo (password);
    echo pi;
?>
<?php
    define("DBServer", "localhost");
    define("DBCatalog", "php_online");
    define("DBUser", "root");
    echo DBServer;
?>
```

Системні константи (вбудовані)

- __LINE__ номер поточного рядку у поточному файлі.
- __FILE__ повне ім`я файла
- __FUNCTION__ ім`я функції
- __CLASS__ поточний клас
- __METHOD__ ім`я поточного метода поточного класса
- PHP_VERSION

Перевизначення типів

```
<?php
$someVar = 123;
echo gettype($someVar);
settype($someVar, "string")
echo gettype($someVar);
?>
```

Функції

- **isset() – перевіряє, чи була оголошена змінна, відмінна від NULL-типу;**
- **empty() - аналог isset(), перевіряє, пуста змінна чи ні. Змінна вважається пустою, якщо вона не існує, або її значення FALSE.**
- **unset() - видаляє змінну і її значення**

Оператор контролю помилок (повідомлення про помилки, згенеровані виразом, будуть проігнорировані)

```
<?php
$file1 = file_get_contents('counter.dat');
$file2 = @file_get_contents('counter.dat');
?>
```

Оператор контролю помилок

(повідомлення про помилки, згенеровані виразом, будуть проігнорировані)

```
<?php
$file1 = file_get_contents('counter.dat');
$file2 = @file_get_contents('counter.dat');
?>

$my_file = @file ('non_existent_file') or
    die ("Помилка при відкритті файла");

@header("Location: ". $_SERVER["REQUEST_URI"]);
//редирект
```

Операція присвоювання

```
$sentence = "My favorite food is $food";
$sentence2 = 'My favorite food is $food';
```

```
<?php
$output = `ipconfig /all`;
echo "$output";
?>
```

Використання лапок

```
<?php
$juice = 'plum';
echo "I drank some juice made of $juices";
// $juices не визначена
echo "I drank some juice made of {$juice}s";
// $juice буде аналізована
$juice = array('apple', 'orange', 'plum');
echo "I drank some juice made of {$juice[1]}s";
// $juice[1] буде аналізована. ?>
```

Фігурний синтиаксис

```
<?php
        error_reporting(E_ALL);
        // Виведення помилок (E_ERROR | E_WARNING|E_PARSE)

        $great = ‘World!’;

        echo “Hello, {$great}";
        echo " Hello,  ${great}";
        echo “Довжина дорівнює  {$square->width}00 сантиметрів.";

        // Ключі у лапках працюють лише з  синтаксисом фігурних лапок
        echo "Значення: {$arr['key']}";

        echo ”Значення  елемента масива: {$arr[4][3]}";
?>
```

Керуючі послідовності

Таблиця Керуючі послідовності

Керуюча послідовність	Значення
\a	Дзвоник
\b	Повернення на крок
\e	Символ ESC
\f	Переведення формату
\n	Перехід на новий рядок
\r	Повернення каретки
\t	Горизонтальна табуляція
\v	Вертикальна табуляція
\$	Знак долару
\@	Комерційне АТ
\0nnn	Вісімковий код символу
\xnn	16-ий код символу
\cn	Емуляція натиснення Ctrl+n, наприклад \cZ Ctrl+Z
\l	Переводить наступний символ у нижній регістр
\u	Переводить наступний символ у верхній регістр
\L	Переводить слідуючу послідовність символів, обмежену керуючою послідовністю \E, у нижній регістр
\Q	В наступній послідовності символів, обмеженій керуючою послідовністю \E,перед кожним неалфавітно-цифровим символом вставляє зворотну косу риску.
\U	Переводить слідуючу послідовність символів, обмежену керуючою послідовністю \E, у верхній регістр
\E	Обмежує дію керуючих послідовностей \L, Q і U
\\	Зворотна коса риска
\”	Подвійні лапки
\’	Одинарні лапки

Оператори

Обозначение	Название	Пример
+	Сложение	$a + $b
-	Вычитание	$a - $b
*	Умножение	$a * $b
/	Деление	$a / $b
%	Остаток от деления	$a % $b

.	Конкатенация (сложение строк)	$c = $a . $b (это строка, состоящая из $a и $b)

=	Присваивание	Переменной слева от оператора будет присвоено значение, полученное в результате выполнения каких-либо операций или переменной/константы с правой стороны	$a = ($b = 4) +5; ($a будет равна 9, $b будет равна 4)
+=		Сокращение. Прибавляет к переменной число и затем присваивает ей полученное значение	$a += 5; (эквивалентно $a = $a + 5;)
.=		Сокращенно обозначает комбинацию операций конкатенации и присваивания (сначала добавляется строка, потом полученная строка записывается в переменную)	$b = "Привет "; $b .= "всем"; (эквивалентно $b = $b . "всем";) В результате: $b="Привет всем"

Операція конкатенації

Наприклад,

```
<?php
        $a = "Hello ";
        $b = $a . "World!";
        //$b містить рядок ”Hello World!”
        $a = "Hello ";
        $a .= "World!";     // $a - "Hello World!"
?>
```

```
<?php
        $var = "hello";
        $world = "world";
        echo "$var" . '$world'; //outputs hello$world
        echo "$var" . "$world"; //outputs helloworld
        echo "$var" .  $world; //outputs helloworld
?>
```

Наприклад,

```
<?php
        echo "thr"."ee";     //prints the string "three"
        echo "twe" . "lve";   //prints the string "twelve"
        echo 1 . 2;           //prints the string "12"
        echo 1.2;             //prints the number 1.2
        echo 1+2;              //prints the number 3

?>
```

Логічні оператори

Обозначение	Название	Описание	Пример
and	И	$a и $b истинны (True)	$a and $b
&&	И		$a && $b
or	Или	Хотя бы одна из переменных $a или $b истинна (возможно, что и обе)	$a or $b
||	Или		$a || $b
xor	Исключающее или	Одна из переменных истинна. Случай, когда они обе истинны, исключается	$a xor $b
!	Инверсия (NOT)	Если $a=True, то !$a=False и наоборот	! $a

Оператори порівняння

з **PHP7 - <=>** Комбінований оператор порівняння

-1, якщо лівий операнд менший правого,

0, якщо вони рівні

1, якщо лівий більший правого.

Обозначение	Название	Пример	Описание
==	Равенство	Значения переменных равны	$a == $b
===	Эквивалентность	Равны значения и типы переменных	$a === $b
!=	Неравенство	Значения переменных не равны	$a != $b
<>	Неравенство		$a <> $b
!==	Неэквивалентность	Переменные не эквивалентны	$a !== $b
<	Меньше		$a < $b
>	Больше		$a > $b
<=	Меньше или равно		$a <= $b
>=	Больше или равно		$a >= $b

Наприклад,

```
<?php
        $a = 100;
        $b = '100';
        var_dump($a == $b); // bool(true)  - порівнюються значення
        var_dump($a === $b); // bool(false)
        var_dump($a == 100); // bool(true)
        var_dump($a == '100'); // Порівнюються значення (ігноруються
        //типи)- bool(true)
        var_dump($a === 100); // Порівнюються типи і значення
        //(integer vs. integer); bool(true)
        var_dump($a === '100'); // Порівнюються типи і значення
        //(integer vs. string); bool(false)
?>
```

Наприклад,

```
<?php
        var_dump(0 == "a"); // 0 == 0 -> true
        var_dump("1" == "01"); // 1== 1 -> true
        var_dump("10" == "1e1"); // 10==10 -> true
        var_dump(100 == "1e2"); // 100==100->true
?>
```

Оператори інкремента, декримента

Обозначение	Название	Описание	Пример
++$a	Пре-инкремент	Увеличивает $a на единицу и возвращает $a	<? $a=4; echo "Должно быть 4:" .$a++; echo "Должно быть 6:" .++$a; ?>
$a++	Пост-инкремент	Возвращает $a, затем увеличивает $a на единицу	
--$a	Пре-декремент	Уменьшает $a на единицу и возвращает $a	
$a--	Пост-декремент	Возвращает $a, затем уменьшает $a на единицу	

```
$a = 1000;
echo $a++; // виведе 1000
echo $a++; // виведе 1001
echo $a--; // виведе 1002
```

Вбудований документ (Heredoc)

```
<?php
$str = <<<EOD
Приклад строки з декількома
рядками з використанням heredoc-синтаксиса
EOD;
// Ініціалізується <<<, змінні інтерпретуються
$name = 'Web-програмування';
echo <<<EOT
Дисципліна "\'$name\' ".
EOT;
?>
```

Nowdoc синтаксис

```
<?php
$str = <<<'EOD'
// Ініціалізується <<<, змінні не інтерпретуються
Example of string
spanning multiple lines
using nowdoc syntax.
$a does not parse.
EOD;

// закривається 'EOD' (на новому рядку без відступів).
```

- Зведення типів
- <?php
  $foo = 10; // ціле число
  $bar = (boolean) $foo;
- ?>

Допускаються наступні зведення типів:

- (int), (integer) - до integer
- (bool), (boolean) - до boolean
- (float), (double), (real) - до float
- (string) - до string
- (array) - до array
- (object) - до object
- (unset) - до NULL

Рядковий тип

```
$sequence_number = "04efgh";
$letter = $sequence_number[4];
```

Перетворення в рядок:

- автоматично (echo(), print(), порівняння),
- масиви перетворюються в *Array* (потрібно використовувати $ar[3])
- *NULL* –пустий рядок
- об'єкти – *Object* (слід використовувати спец ф-ї, наприклад, _toString)
- Ресурси - *"Resource id #1"*, де *1* - унікальний номер ресурса (resource), наданий PHP в момент виконання. (існує функція get_resource_type()).

- зведення *(string),*
- функція *strval()* – повертає рядкове значення змінної,

Приклад,

```
<?php
    $foo = 10;          // $foo ціле число
    $str = "$foo";      // $str - рядок
    $fst = (string) $foo; // $fst - рядок

    if ($fst === $str) {
        echo "вони однакові";
    }
    ?>
```

Перетворення рядків в числа

Якщо рядок містить '.', 'e', або'E', то –integer
інакше – float

- ```
  <?php
  $foo = 1 + "10.5";                // float (11.5)
  $foo = 1 + "-1.3e3";              // float (-1299)
  $foo = 1 + "bob-1.3e3";           // integer (1)
  $foo = 1 + "bob3";                // integer (1)
  $foo = 1 + "10 Small Pigs";       // integer (11)
  $foo = 4 + "10.2 Little Piggies"; // float (14.2)
  $foo = "10.0 pigs " + 1;          // float (11)
  $foo = "10.0 pigs " + 1.0;        // float (11)
  ?>
  ```
  ```

Доступ до символів в рядку

- <?php

  $str = 'This is a test.';
  $first = $str[0]; // перший символ, також можна - *$str{0}*.

  $third = $str[2]; // третій символ

  $str = 'This is still a test.';
  $last = $str[strlen($str)-1]; // останній символ

  $str = 'Look at the sea';
  $str[strlen($str)-1] = 'e'; // зміна символа
  ?>

Функції для роботи з рядками

strlen(string $st) – довжина рядків

```
$x = "Hello!";
echo strlen($x); // Виводить 6
```

strpos(string $where, string $what, int $fromwhere=0) – наявність підрядку

substr(string $str, int $start [,int $length])

```
$str = "Programmer";
echo substr($str,0,2); // Виводить Pr
echo substr($str,-3,3); // Виводить mer
```

strcmp(string $str1, string $str2) - порівняння перших симв.

strcasecmp(string $str1, string $str2) –порівн. перших симв.без урахування регистру

str_replace(string $from, string $to, string $str)

```
$st=str_replace("\n","<br>\n",$str)
```

str_word_count() – підраховує кількість слів

```
<?php
echo str_word_count("Hello world!"); // outputs 2
?>
```

strrev() – реверс символів

```
<?php
echo strrev("Hello world!"); // outputs !dlrow olleH
?>
```

explode() – розбиває рядок за роздільником

```
<?php
$str = "Hello world. It's a beautiful day.";
print_r (explode(" ",$str));
?>
```

wordwrap(string $str, int $width=75, string $break="\n") –
перенесення рядків

```
$str = WordWrap ($str, 20, "<br>");
echo $str;
```

strip_tags (string $str [, string $allowable_tags])

```
$stripped = strip_tags ($str); // Видаляє всі html
    $stripped = strip_tags($str, "<head><title>");
// Видаляє всі html - теги, окрім html-тегів <head> і <title>
```

trim(string $str) – видаляє пробіли з початку і з кінця рядка

ltrim(string $st) –з початку,

chop(string $st) - **rtrim(string $st)**– з кінця рядка

strtoupper(string $str) – регістр

strtolower(string $str))

strtr(string $str, string $from, string $to) –заміна символів

Дякую за увагу!

Подивитись