#### Лекция 7

#### Тема: Тип array (массив)

***Массив*** в PHP представляет собой упорядоченную карту – *тип*, который преобразует *значения* в *ключи*. Этот *тип* оптимизирован в нескольких направлениях, поэтому вы можете использовать его как собственно *массив*, список (вектор), хеш-таблицу (являющуюся реализацией карты), стек, очередь и т.д. Поскольку вы можете иметь в качестве *значения* другой *массив* PHP, можно также легко эмулировать деревья.

Определить *массив* можно с помощью конструкции *array* () или непосредственно задавая *значения* его элементам.

##### Определение при помощи array()

array (key => value,

key1 => value1, ... )

Языковая конструкция ***array*** () принимает в качестве параметров пары *ключ* => ***значение***, разделенные запятыми. Символ => устанавливает соответствие между *значением* и его *ключом*. *Ключ* может быть как *целым числом*, так и *строкой*, а *значение* может быть любого имеющегося в PHP *типа*. Числовой *ключ**массива* часто называют индексом. Индексирование *массива* в PHP начинается с нуля. *Значение* элемента *массива* можно получить, указав после имени *массива* в *квадратных скобках**ключ* искомого элемента. Если ***ключ****массива* представляет собой стандартную запись *целого числа*, то он рассматривается как число, в противном случае – как *строка*. Поэтому запись $a["1"] равносильна записи $a[1], так же как и $a["-1"] равносильно $a[-1].

<?php

$books = array ("php" =>

"PHP users guide",

12 => true);

echo $books["php"];

//выведет "PHP users guide"

echo $books[12];

//выведет 1

?>

Пример 2.9. Массивы в PHP

Если для элемента *ключ* не задан, то в качестве *ключа* берется максимальный числовой *ключ*, увеличенный на единицу. Если указать *ключ*, которому уже было присвоено какое-то *значение*, то оно будет перезаписано. Начиная с PHP 4.3.0, если максимальный *ключ* – отрицательное число, то следующим *ключом**массива* будет ноль ( 0 ).

<?php

// массивы $arr и $arr1 эквивалентны

$arr = array(5 => 43, 32, 56, "b" => 12);

$arr1 = array(5 => 43, 6 => 32,

7 => 56, "b" => 12);

?>

Пример 2.10. Массивы в PHP

Если использовать в качестве *ключа* TRUE или FALSE, то его *значение* переводится соответственно в единицу и ноль *типа integer*. Если использовать *NULL*, то вместо *ключа* получим пустую *строку*. Можно использовать и саму пустую *строку* в качестве *ключа*, при этом ее надо брать в кавычки. Так что это не то же самое, что использование пустых *квадратных скобок*. Нельзя использовать в качестве *ключа* *массивы* и *объекты*.

##### Определение с помощью синтаксиса квадратных скобок

Создать *массив* можно, просто записывая в него *значения*. Как мы уже говорили, *значение* элемента *массива* можно получить с помощью *квадратных скобок*, внутри которых нужно указать его *ключ*, например, $book["php"]. Если указать новый *ключ* и новое *значение*, например, $book["new\_key"]="new\_value", то в *массив* добавится новый элемент. Если мы не укажем *ключ*, а только присвоим *значение* $book[]="new\_value", то новый элемент *массива* будет иметь числовой *ключ*, на единицу больший максимального существующего. Если *массив*, в который мы добавляем *значения*, еще не существует, то он будет *создан*.

<?

$books["key"]= value; // добавили в массив

// $books значение

// value с ключом key

$books[] = value1; /\* добавили в массив

значение value1 с

ключом 13, поскольку

максимальный ключ у

нас был 12 \*/

?>

Для того чтобы изменить конкретный элемент *массива*, нужно просто присвоить ему с его *ключом* новое *значение*. Изменить *ключ* элемента нельзя, можно только *удалить элемент*(пару *ключ* / *значение* ) и добавить новую. Чтобы ***удалить элемент****массива*, нужно использовать функцию ***unset*** () .

<?php

$books = array ("php" =>

"PHP users guide",

12 => true);

$books[] =

"Book about Perl"; // добавили элемент

// с ключом (индексом)

// 13 это эквивалентно

// $books[13] =

// "Book about Perl";

$books["lisp"] =

123456; /\* Это добавляет к массиву новый

элемент с ключом "lisp" и

значением 123456 \*/

unset($books[12]); // Это удаляет элемент

// c ключом 12 из массива

unset ($books); // удаляет массив полностью

?>

Заметим, что, когда используются пустые *квадратные скобки*, максимальный числовой *ключ* ищется среди *ключей*, существующих в *массиве* с момента последнего *переиндексирования*. ***Переиндексировать массив*** можно с помощью функции ***array\_values*** ().

<?php

$arr =

array ("a","b","c"); /\* Создаем массив

со значениями

"a", "b" и "c".

Поскольку ключи

не указаны, они

будут 0,1,2

соответственно \*/

print\_r($arr); // выводим массив (и ключи,

// и значения)

unset($arr[0]);

unset($arr[1]);

unset($arr[2]);

// удаляем из него все значения

print\_r($arr); // выводим массив (и ключи,

// и значения)

$arr[] = "aa"; // добавляем новый элемент

// в массив.

// Его индексом (ключом)

// будет 3, а не 0

print\_r($arr);

$arr =

array\_values($arr); // переиндексируем

// массив

$arr[] = "bb"; // ключом этого элемента

// будет 1

print\_r($arr);

?>

Пример 2.11. Переиндексация массива

Результатом работы этого скрипта будет:

Array ( [0] => a [1] => b [2] => c )

Array ( )

Array ( [3] => aa )

Array ( [0] => aa [1] => bb )

#### Тип object (объекты)

***Объекты*** – *тип данных*, пришедший из объектно-ориентированного программирования (*ООП*). Согласно принципам *ООП*, класс – это набор *объектов*, обладающих определенными свойствами и методами работы с ним, а *объект* соответственно – экземпляр класса. Например, программисты – это класс людей, которые пишут программы, изучают компьютерную литературу и, кроме того, как все люди, имеют имя и фамилию. Теперь, если взять одного конкретного программиста, Васю Иванова, то можно сказать, что он является *объектом* класса программистов, обладает теми же свойствами, что и другие программисты, тоже имеет имя, пишет программы и т.п.

В PHP для доступа к методам *объекта* используется *оператор* ->. Для инициализации *объекта* используется выражение new, создающее в *переменной* экземпляр *объекта*.

<?php

//создаем класс людей

class Person

{

// метод, который обучает человека PHP

function know\_php()

{

echo "Теперь я знаю PHP";

}

}

$bob = new Person; // создаем объект

// класса человек

$bob -> know\_php(); // обучаем его PHP

?>

Пример 2.12. Объекты в PHP

Более подробно реализацию принципов *ООП* в языке PHP мы рассмотрим в одной из следующих лекций.

#### Тип resource (ресурсы)

***Ресурс*** – это специальная *переменная*, содержащая ссылку на внешний *ресурс* (например, соединение с базой данных). *Ресурсы* создаются и используются специальными функциями (например, *mysql\_connect*(), pdf\_new() и т.п.).

#### Тип Null

Специальное значение *NULL* говорит о том, что *переменная* не имеет значения.

*Переменная* считается *NULL*, если:

* ей была присвоена *константа* *NULL* ( $var = NULL );
* ей еще не было присвоено какое-либо значение;
* она была удалена с помощью *unset* ().

Существует только одно значение *типа**NULL* – регистронезависимое ключевое слово NULL.

### Решение задачи

Теперь вернемся к задаче, которую мы поставили в самом начале лекции. Напомним, что она состояла в *составлении письма* разным людям по поводу разных событий. Попытаемся использовать для решения этой задачи изученные средства – *переменные*, *операторы*, *константы*, *строки* и *массивы*. В зависимости от получателя изменяется событие и обращение, указанные в письме, поэтому естественно вынести эти величины в *переменные*. Более того, поскольку событий и людей много, удобно использовать *переменные* *типа**массив*. Подпись в письме остается постоянной всегда, поэтому логично задать ее как *константу*. Чтобы не писать слишком длинные и громоздкие *строки*, используем *оператор* *конкатенации*. Итак, вот что получилось:

<?

// пусть наша подпись

// будет константой

define("SIGN","С уважением, Вася");

// зададим массивы людей и событий

$names = array("Иван Иванович",

"Петр Петрович",

"Семен Семенович");

$events = array(

"f" => "день открытых дверей",

"o" => "открытие выставки",

"p" => "бал выпускников");

// составим текст приглашения

$str = "Уважаемый (ая), $names[0].";

$str .= "<br>Приглашаем Вас на ".

$events["f"];

$str .= "<br>" . SIGN;

echo $str; // выведем текст на экран

?>

### Заключение

Итак, в этой лекции мы познакомились с *основами синтаксиса* языка *PHP*, научились работать с *переменными* различных *типов*, *константами*, *операторами*, познакомились со всеми существующими в *PHP* *типами данных*. Говоря о таких *типах данных*, как *строки* и *массивы*, мы разобрали только самые основы. Эти конструкции настолько удобны и просты в использовании, что заслуживают отдельного разговора. Подробности будут изложены в последующих лекциях. Решая задачу, мы старались использовать только имеющиеся знания и не забегать вперед, поэтому решение не слишком подходит для практического применения. В следующей лекции мы исправим этот недостаток и создадим более универсальный *шаблон* электронного письма.