Internet-технологии

ЛЕКЦИЯ №8
ОСНОВЫ ЯЗЫКА PHP

Содержание

- Общие сведения про язык PHP.
- Основы синтаксиса. Переменные, типы данных, управляющие структуры.
- Глобальные константы.
- Массивы, функции для работы с массивами.

Введение в PHP

PHP (PHP: Hypertext Preprocessor) – скриптовый язык программирования, предназначенный для генерации HTML-страниц на веб-сервере.

June 1995 – PHP Tools 1.0 (Personal Home Page Tools)
April 1996 – PHP/FI
June 1996 – PHP/FI 2.0 (beta)
November 1997 – PHP/FI 2.0

June 1998 – PHP 3.0 (PHP: Hypertext Preprocessor)
Zend Technologies
May 2000 – PHP 4.0
July 2004 – PHP 5
Nov 12 2015 – PHP 7 Final

Расмус Лердорф
«**PHP** – Tools for **Personal Home Page**»

Введение в PHP

Области применения:

- Создание скриптов для выполнения **на стороне сервера**. (LAMP).
- Создание скриптов для выполнения в командной строке.
- Создание оконных приложений, выполняющихся на стороне клиента (расширение PHP-GTK).

Возможности:

- выбор между использованием процедурного или объектно-ориентированного программирования;
- генерация XML;
- работа с cookies и сессиями;
- возможности обработки текста, включая регулярные выражения;

Сборки веб-серверов для разработки

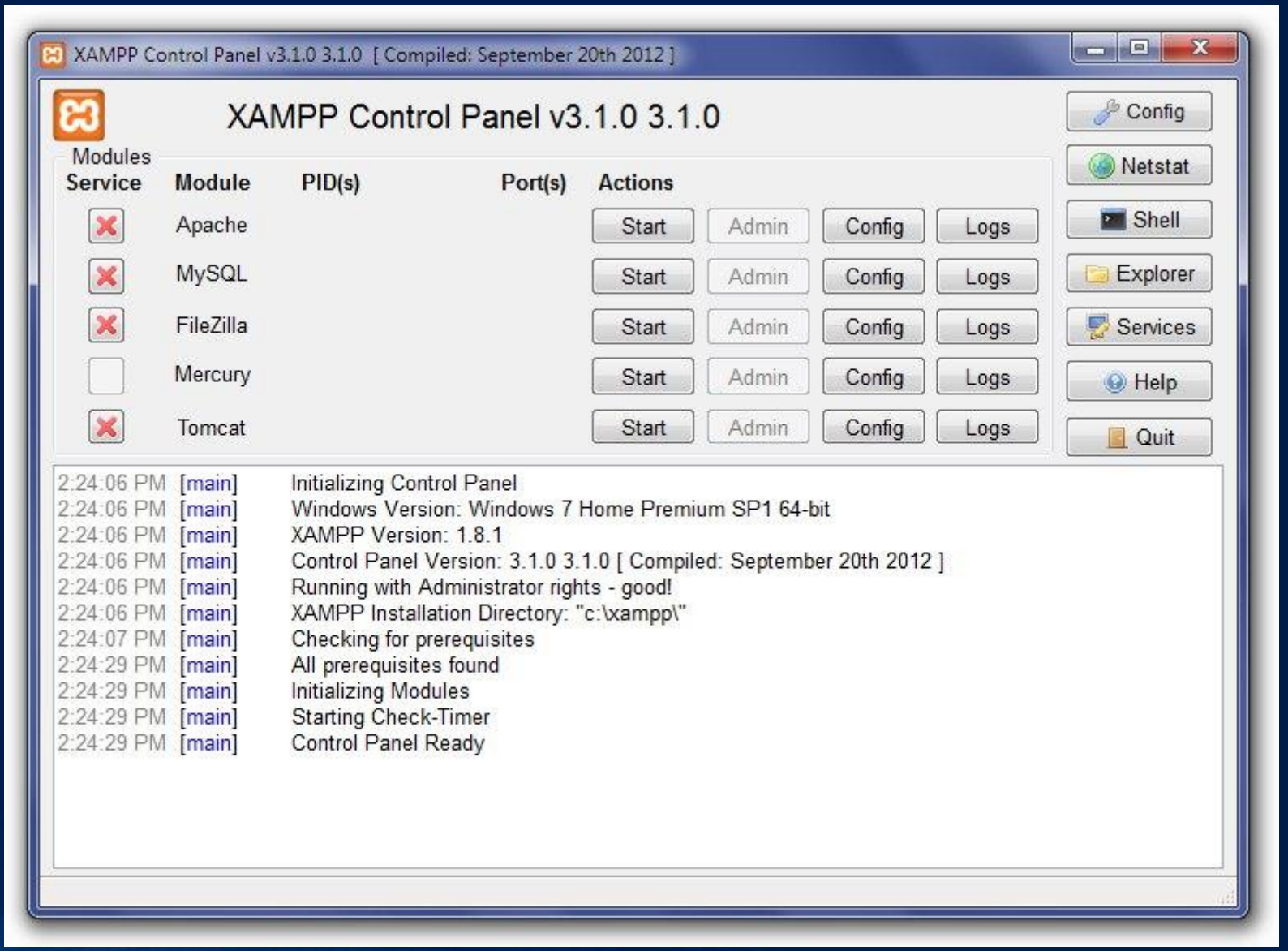
Web-сервер является программой, которая обеспечивает доставку контента конечному пользователю по сети. Реализует серверную часть протокола HTTP, принимая HTTP-запросы от клиентов и возвращая им HTTP-ответы, как правило, вместе с HTML-страницей, изображениями, файлами и т.д.

Готовые сборки веб-серверов, объединяющие все необходимые компоненты для работы (Apache, MySQL, PHP…):

- XAMPP
- Denwer (от сокр. Д.н.w.р. – джентльменский набор Web-разработчика)
- TopServer

Сборки веб-серверов для разработки. XAMPP

XAMPP:
X – все ОС,
A – Apache,
M – MySQL,
P – PHP,
P – Perl.



Пример

Test2.php:

```
<!DOCTYPE HTML>
<html>
<head>
 <title>Тест</title>
 <meta charset="utf-8">
</head>
<body>
Тест 1:
<?php
echo "Hello, World!<br>";
?>
Тест 2:
<?php echo 2+2 ?>
</body>
</html>
```

http://localhost/test2.php

Тест 1: Hello, World!
Тест 2: 4

Исходный код сгенерированной страницы:

```
<!DOCTYPE HTML>
<html>
<head>
 <title>Тест</title>
 <meta charset="utf-8">
</head>
<body>
Тест 1:
Hello, World! <br> Тест 2:
4 </body>
</html
```

Размещение кода на HTML-странице

Способы вставки PHP-кода: **<?php** … ?>, **<?** … ?>.

Все, что находится вне пары открывающегося и закрывающегося тегов, игнорируется интерпретатором PHP. Если файл содержит только код PHP, предпочтительно опустить закрывающий тег в конце файла.

<?php phpinfo(); **?>**
<? phpinfo(); **?>** (если в php.ini short_open_tag = On)
<?= "Hello" ?> Это синоним для "<? echo "Hello" ?>"

Устаревшие способы:
<% … %>
<%= … %>
<script language="php"> … </script>

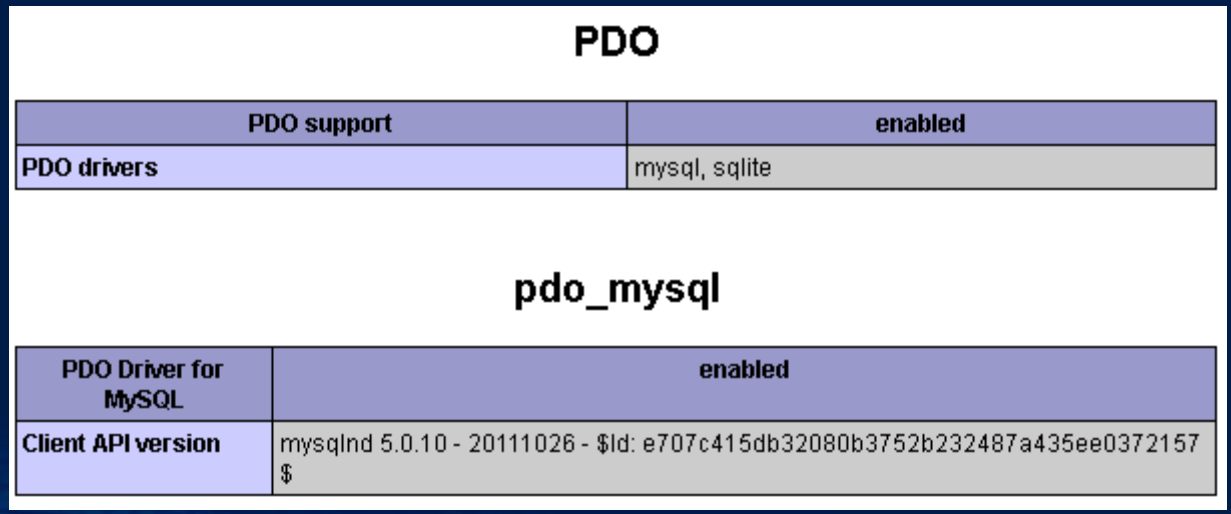
```
<?php
if($expression) {
?>
<b>Это истина.</b>
<?php
} else {
?>
<b>Это ложь.</b>
<?php
}
?>
```

Функция phpinfo

phpinfo – выводит большое количество информации о текущем состоянии PHP: о настройках, о расширениях, о версии, информация о сервере, версии ОС, о путях, значениях настроек конфигурации, о HTTP заголовках и лицензии PHP.

Возвращает TRUE в случае успешного завершения или FALSE в случае возникновения ошибки. Вывод функции можно настраивать, передавая в качестве необязательного аргумента битовую маску из одной или более приведенных ниже констант.

INFO_GENERAL	1
INFO_CREDITS	2
INFO_CONFIGURATION	4
INFO_MODULES	8
INFO_ENVIRONMENT	16
INFO_VARIABLES	32
INFO_LICENSE	64
INFO_ALL	-1



PDO

PDO support	enabled
PDO drivers	mysql, sqlite

pdo_mysql

PDO Driver for MySQL	enabled
Client API version	mysqlnd 5.0.10 - 20111026 - $Id: e707c415db32080b3752b232487a435ee0372157 $

Синтаксис PHP

Имена переменных начинаются с символа **$**, тип переменной объявлять не нужно.

Имена переменных и констант чувствительны к **регистру** символов. Имена классов, методов классов и функций к регистру символов не чувствительны.

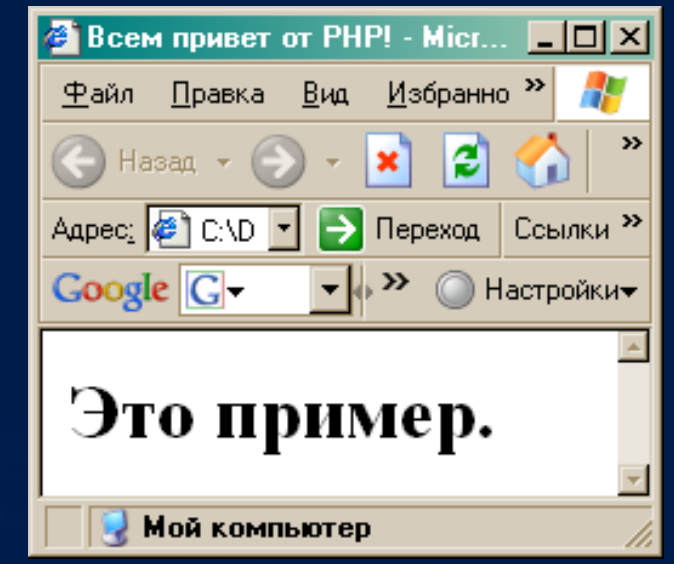
Переменные обрабатываются в строках, заключённых в двойные кавычки, и heredoc-строках (строках, созданных при помощи оператора <<<). Переменные в строках, заключённых в одинарные кавычки, не обрабатываются.

Инструкции разделяются с помощью **точки с запятой**;

PHP поддерживает три типа комментариев: в стиле языка Си (ограниченные **/* */**), C++ (начинающиеся с **//** и идущие до конца строки) и оболочки UNIX (с **#** до конца строки).

Синтаксис PHP. Комментарии

```
<?php
  echo "Это тест"; // Это однострочный комментарий в стиле c++
  /* Это многострочный комментарий
     еще одна строка комментария */
  echo "Это еще один тест";
  echo "Последний тест"; # Это комментарий в стиле Unix
?>
<h1>Это <?php # echo "простой";?> пример.</h1>
```



Основы синтаксиса. Типы данных

PHP является языком программирования с динамической типизацией, не требующим указания типа при объявлении переменных, равно как и самого объявления переменных.

PHP поддерживает восемь простых типов:

Четыре скалярных типа:

- логический boolean
- целочисленный integer: 1234; -123; 0123; 0x1A; 0b11111111;
- вещественный float (или 'double'): 1.234; 1.2e3;
- строковый string

Два смешанных типа:

- array
- object

Два специальных типа:

- resource
- NULL

некоторые псевдотипы:

- любой тип mixed
- число (integer либо float) number
- отсутствие параметров void

Основы синтаксиса. Типы данных. gettype()

Выражение	gettype()
$x = "";	string
$x = null;	NULL
$x не определена	NULL
$x = array();	array
$x = false;	boolean
$x = true;	boolean
$x = 1;	integer
$x = 42;	integer
$x = 4e2;	double

Выражение	gettype()
$x = 4.2;	double
$x = 0;	integer
$x = -1;	integer
$x = "1";	string
$x = "0";	string
$x = "-1";	string
$x = "php";	string
$x = "true";	string
$x = "false";	string

Основы синтаксиса. Типы данных. bool

Для указания логического значения используются **регистронезависимые** константы TRUE или FALSE. При преобразовании в логический тип число 0, пустая строка, ноль в строке «0», NULL и пустой массив считаются равными FALSE. Все остальные значения автоматически преобразуются в TRUE.

При преобразовании из булева типа FALSE преобразуется в 0 (ноль), а TRUE – в 1 (единицу).

```
<?php
  var_dump((bool) "");       // bool(false)
  var_dump((bool) 1);        // bool(true)
  var_dump((bool) -2);       // bool(true)
  var_dump((bool) "foo");  // bool(true)
  var_dump((bool) 2.3e5);    // bool(true)
  var_dump((bool) array(12)); // bool(true)
  var_dump((bool) array());   // bool(false)
  var_dump((bool) "false");   // bool(true)
?>
```

Основы синтаксиса. Типы данных. string

Строка – это набор символов. Строка может быть определена четырьмя различными способами:

- **одинарными кавычками** (вывод как есть, кроме \)
- **двойными кавычками** (PHP распознает управляющие последовательности для специальных символов);
- heredoc-синтаксисом: <<<ID строка ID;
- nowdoc-синтаксисом: <<<'ID' строка ID;

```
Значение переменной: 5
Значение переменной: $x
```

```
<?php
 $x=5;
 $str = <<<ID
 Значение переменной: $x
 ID;
 echo $str."<br>";
```

```
$str = <<<'ID'
 Значение переменной: $x
 ID;
 echo $str;
?>
```

Основы синтаксиса. Определение строки

```php
<?php
echo 'Переводы строки
не обрабатываются';
echo 'Пример \'использования одинарных
кавычек\'';
echo 'Переноса строки здесь \n нет';
echo 'Значения переменных (например, $x) не
являются частью строки';
echo '<br>';
?>
```

```
Переводы строки не обрабатываются
Пример 'использования одинарных кавычек'
Переноса строки здесь \n нет
Значения переменных (например, $x) не являются частью строки
```

Основы синтаксиса. Определение строки

При определении с помощью двойных кавычек доступные управляющие последовательности:

последовательность	значение
\n	новая строка (LF или 0x0A (10) в ASCII)
\r	возврат каретки (CR или 0x0D (13) в ASCII)
\t	горизонтальная табуляция (HT или 0x09 (9) в ASCII)
\\	обратная косая черта
\$	знак доллара
\"	двойная кавычка

```
<?php
echo "Переноса строки здесь \n все равно нет";
echo nl2br("Перенос строки здесь \n есть");
echo "<br>";
?>
```

```
Переноса строки здесь все равно нет
Перенос строки здесь
есть
```

Основы синтаксиса. Вывод строки. Пример

```
<?php
$beer = 'Heineken';

echo "He drank some $beer beer.";
echo "He drank some $beers";          // error
echo "He drank some ${beer}s";        //He drank some Heinekens
echo "He drank some {$beer}s";        //He drank some Heinekens
echo "He drank some { $beer}s";       //He drank some { Heineken}s
?>
```

Любая скалярная переменная, элемент массива или свойство объекта, отображаемое в строку, может быть представлена в строке фигурным синтаксисом – запишите выражение так же, как и вне строки, а затем заключите его в { и }. Поскольку { не может быть экранирован, этот синтаксис будет распознаваться только когда $ следует непосредственно за {.

Основы синтаксиса. Доступ к символу строки

Символы в строках можно использовать и модифицировать, определив их смещение относительно начала строки, начиная с нуля, в квадратных скобках после строки, например, $str[42].

```
<?php
$str = 'This is a test.';
$first = $str[0];
$third = $str[2]; // Получение последнего символа строки
$str = 'This is still a test.';
$last = $str[strlen($str)-1];
// Изменение последнего символа строки
$str = 'Look at the sea';
$str[strlen($str)-1] = 'e';
?>
```

```
This is a test.
T
i

Look at the see
```

Основы синтаксиса. Функции для работы со строками

htmlentities — Преобразует все возможные символы в соответствующие HTML-сущности;

join — Объединяет элементы массива в строку;

nl2br — Вставляет HTML-код разрыва строки перед каждым переводом строки;

ord — Возвращает ASCII-код символа;

parse_str — **Разбирает строку в переменные**;

strlen — Возвращает длину строки;

substr — Возвращает подстроку;

substr_count — Возвращает число вхождений подстроки;

substr_replace — Заменяет часть строки;

trim — Удаляет пробелы (или другие символы) из начала и конца строки.

```
$str = "A 'quote' is <b>bold</b>";
echo htmlentities($str);
// выводит: A 'quote' is &lt;b&gt;bold&lt;/b&gt;
```

Основы синтаксиса. Преобразование в строку

Вы можете преобразовать значение в строку, используя приведение **(string)**, либо функцию **strval()**. В выражениях, где необходима строка (например, при использовании **echo()**, **print()**, сравнении значения переменной со строкой), преобразование происходит **автоматически**.

Булево (boolean) значение TRUE преобразуется в строку "1", а значение FALSE представляется как "" (пустая строка).

Целое (integer) или число с плавающей точкой (float) преобразуется в строку, представленную числом, состоящим из его цифр (включая показатель степени для чисел с плавающей точкой).

Массивы всегда преобразуются в строку "Array".

NULL всегда преобразуется в пустую строку.

Основы синтаксиса. Преобразование строк в числа

Если строка не содержит какой-либо из символов '.', 'e', или 'E', и значение числа помещается в пределы целых чисел (определенных PHP_INT_MAX), строка будет распознана как целое число (integer). Во всех остальных случаях она считается числом с плавающей точкой (float).

```
<?php
$foo = (float)"10.5";                 // float (10.5)
$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)
?>
```

Основы синтаксиса. Переменные

Переменные в PHP начинаются знаком **доллара** с последующим именем переменной. Имя переменной **чувствительно к регистру** и должно начинаться с **буквы** или символа **подчеркивания** и состоять из букв, цифр и символов подчеркивания в любом количестве.

Для присвоения **по ссылке** используется амперсанд (&) перед началом имени переменной (исходной переменной). По ссылке могут быть присвоены **только именованные** переменные.

Переменная переменной берет значение переменной и рассматривает его как имя переменной.

```
$str = 'строка1'; $str2 = &$str;        // Ссылка на $str через $str2.
$str2 = "строка2";
echo $str2;                 //строка2
echo $str;                  //строка2
$bar = &test();             //ошибка!
$$str2 = 'world';  echo $строка2;      // world
```

Основы синтаксиса. Область видимости переменных

Любая используемая внутри функции переменная по умолчанию ограничена **локальной** областью видимости функции. Если **глобальная** переменная будет использоваться внутри функции, она должна быть объявлена глобальной внутри функции. Второй способ доступа к переменным глобальной области видимости – использование специального массива $GLOBALS.

Статическая переменная существует только в локальной области видимости функции, но не теряет своего значения, когда выполнение программы выходит из этой области видимости. Нельзя в качестве значения присваивать результат выражения.

```
$a = 1; /* глобальная область видимости */
function test() {
  echo $a; /* ссылка на переменную локальной области видимости */
  global $a; echo $a;
}
test();
```

Основы синтаксиса. Константы

Константы определяются с помощью функции **define()** или с помощью ключевого слова **const**. После того, как константа определена, ее значение не может быть изменено. В отличие от переменных, не нужно предварять имя константы символом $. Имена констант чувствительны к регистру. По принятому соглашению, имена констант всегда пишутся в верхнем регистре.

Используйте функцию **get_defined_constants()** для получения списка всех объявленных констант.

```
define("FOO",     "something");
define("FOO2BAR", "something more");
define("2FOO",    "something");// Неправильное имя константы
const CONSTANT = 'Здравствуй, мир.';
const ANOTHER_CONST = CONSTANT.'; Прощай, мир.';//после версии 5.6
```

Основы синтаксиса. Массивы

Массив в PHP – это упорядоченное отображение, которое устанавливает соответствие между значением и ключом. Поддерживают числовые и строковые ключи и являются гетерогенными. Массивы могут содержать значения любых типов, включая другие массивы. Порядок элементов и их ключей сохраняется.

```
$array = array(
    "foo" => "bar",
    "bar" => "foo",
);
$array = [
    "foo" => "bar",
    "bar" => "foo",
];
```

```
$array = array(
    1 => "a",
    "1" => "b",
    1.5 => "c",
    true => "d",
);
var_dump($array);
```

```
array(1) { [1]=> string(1) "d" }
```

Если несколько элементов в объявлении массива используют одинаковый ключ, то только последний будет использоваться, а все другие будут перезаписаны.

```
$array = array("foo", "bar", "hello", "world");
```

Основы синтаксиса. Функции для работы с массивами

array_change_key_case — Меняет регистр ключей в массиве
array_combine — Создает новый массив, используя один массив в качестве ключей, а другой в качестве соответствующих значений
array_count_values — Подсчитывает количество всех значений массива
array_flip — Меняет местами ключи с их значениями в массиве
array_keys — Возвращает все или некоторое подмножество ключей массива
array_merge — Сливает один или большее количество массивов
array_pop — Извлекает последний элемент массива
array_rand — Выбирает одно или несколько случайных значений из массива
array_search — Осуществляет поиск данного значения в массиве и возвращает соответствующий ключ в случае удачи
array_unique — Убирает повторяющиеся значения из массива
array_values — Выбирает все значения массива
rsort — Сортирует массив в обратном порядке
shuffle — Перемешивает массив

Функции вывода.

echo — Выводит одну или более строк.
print — Выводит строку. *В чем разница между функциями echo и print?*
string **var_dump**(mixed $expression, bool $return=false)

```
<?php
   $a = array (1, array ('a' , 'b', 'c'));
   var_dump($a);
  echo "<br>";
   print_r($a);
?>
```

```
array(2) { [0]=> int(1) [1]=> array(3) { [0]=> string(1) "a" [1]=> string(1) "b" [2]=> string(1) "c" } }
Array ( [0] => 1 [1] => Array ( [0] => a [1] => b [2] => c ) )
```

Основы синтаксиса. Функции

Синтаксис объявления функции:

```
function foo($arg_1, $arg_2, /* ..., */ $arg_n)
{
   echo "Example function.\n";
   return $retval;
}
```

Внутри функции можно использовать любой корректный PHP-код, в том числе другие функции. Функции не обязаны быть определены до их использования, исключая тот случай, когда функции определяются условно.

Все функции и классы PHP имеют глобальную область видимости – они могут быть вызваны вне функции, даже если были определены внутри и наоборот.

PHP не поддерживает перегрузку функции, также отсутствует возможность переопределить или удалить объявленную ранее функцию.

Альтернативный синтаксис управляющих структур

PHP предлагает альтернативный синтаксис для некоторых его управляющих структур, а именно: if, while, for, foreach и switch. В каждом случае основной формой альтернативного синтаксиса является изменение открывающей фигурной скобки на двоеточие (:), а закрывающей скобки на endif;, endwhile;, endfor;, endforeach; или endswitch; соответственно.

```
<?php if ($a == 5): ?>
A равно 5
<?php endif; if ($a == 5):
    echo "a равно 5";
    echo "...";
elseif ($a == 6):
    echo "a равно 6!!!";
else:
    echo "a не равно ни 5 ни 6";
endif;?>
```

Смешивание синтаксиса в одном и том же блоке управления не поддерживается.

Основы синтаксиса. Суперглобальные массивы

$GLOBALS — Ссылки на все переменные глобальной области видимости.
$_SERVER — Информация о сервере и среде исполнения.
$_**GET** — GET-переменные HTTP.
$_**POST** — POST-переменные HTTP.
$_FILES — Переменные файлов, загруженных по HTTP.
$_REQUEST — Переменные HTTP-запроса.
$_**SESSION** — Переменные сессии.
$_ENV — Переменные окружения.
$_**COOKIE** — HTTP Куки.

```
<form action="foo.php" method="post">
  Имя:  <input type="text" name="username" /><br />
  Email: <input type="text" name="email" /><br />
  <input type="submit" name="submit"/>
</form>
```

```
<?php
echo $_POST['username'];
echo $_REQUEST['email'];
?>
```

Вопросы

- Для чего используется язык PHP?
- Какие готовые сборки веб-серверов вам известны?
- Какие существуют способы вставки кода на HTML-странице?
- Назначение функции phpinfo.
- Определение строковых переменных в языке PHP.
- Какие типы данных поддерживает PHP?
- Что такое heredoc-синтаксис?
- Правила приведения типов.
- Что такое переменные переменные? Какие бывают области видимости?
- Как определяются константы?
- Какие функции для работы с массивами вам известны?
- В чем разница между языковыми конструкциями echo и print?
- В чем отличия альтернативного синтаксиса управляющих структур?
- Какие суперглобальные массивы вам известны?