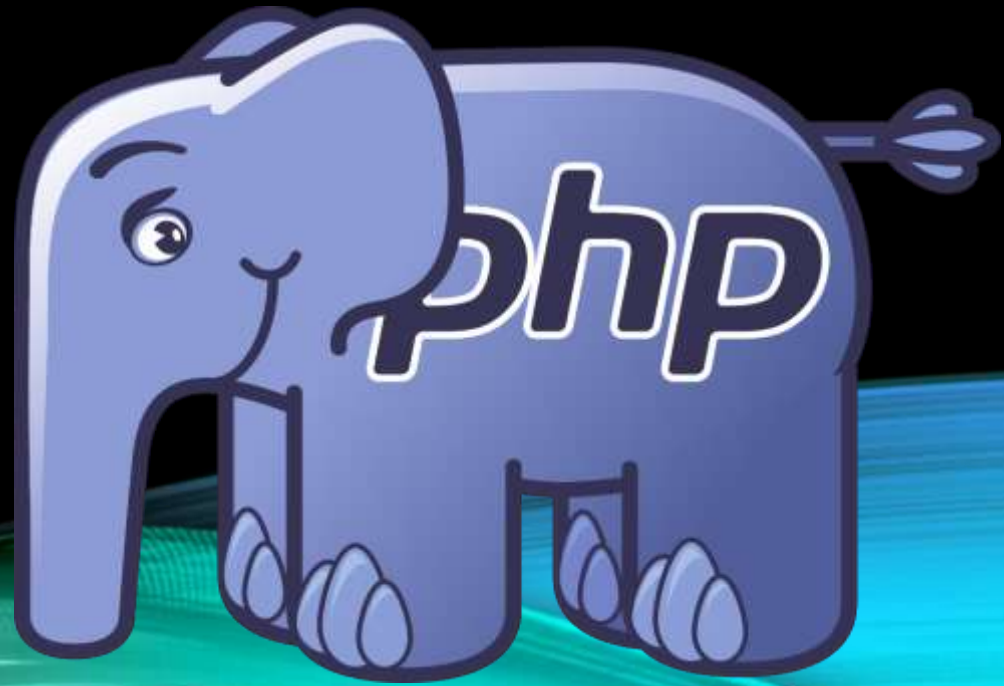
Пикулев В., Фролов А. и Николаев Д., 2018-19

Web-технологии

Тема 4. Язык PHP



ОБЩИЕ СВЕДЕНИЯ О PHP

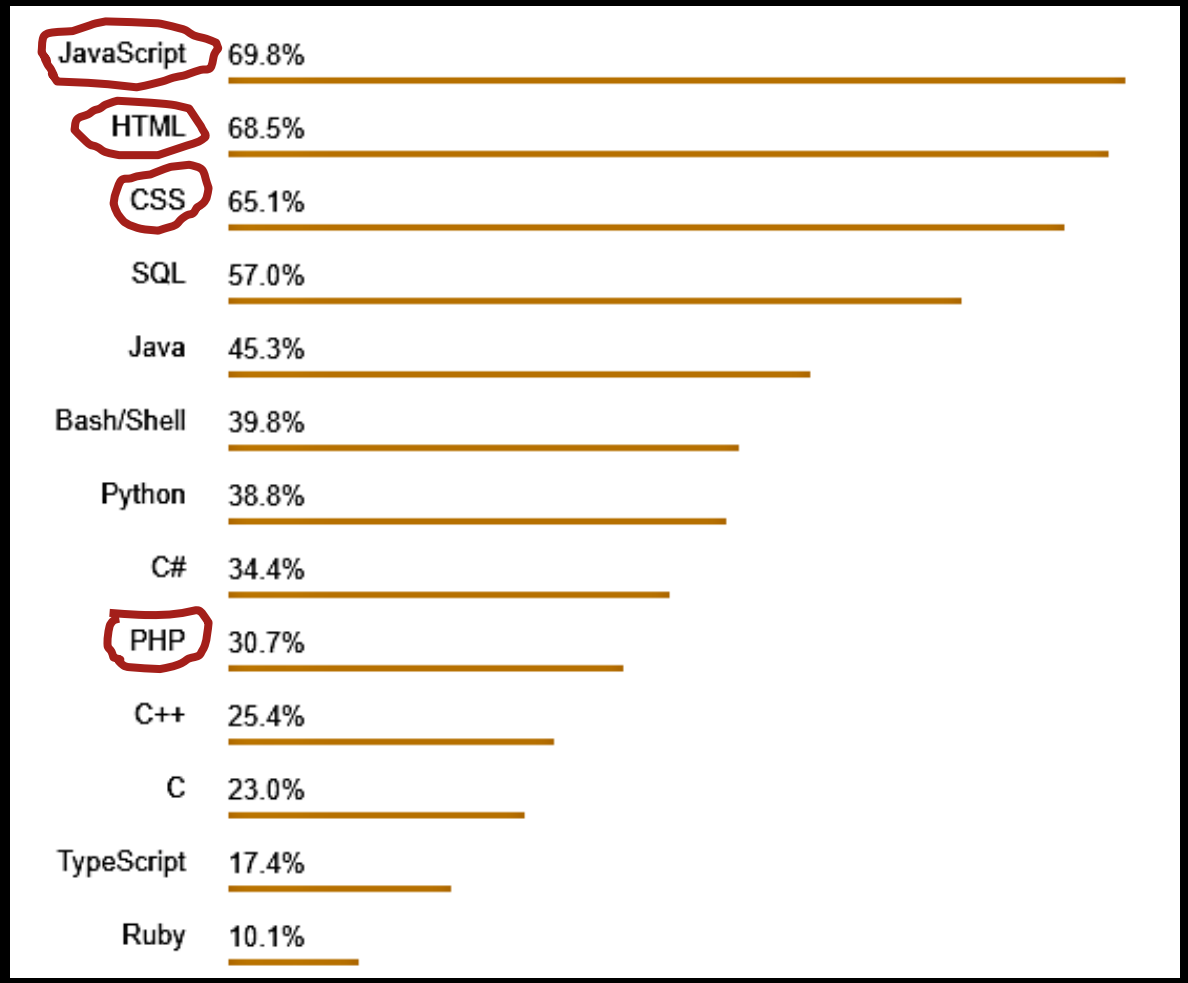
- Области применения
  - Скрипты, исполняемые на стороне сервера (классическое применение)
  - Скрипты, выполняющиеся в командной строке (парсер, отладка)
  - GUI – приложения на стороне клиента (PHP-GTK, необычное применение)

  PHP – мультиплатформенный язык

- Возможности
  - Генерация HTML (динамические web-страницы)
  - Поддержка различных баз данных и различных протоколов доступа к данным (ODBC, сокеты, и пр.)
  - Работа со слабоструктурированными данными (в т.ч. текстовые файлы). Использование регулярных выражений. Генерация XML и PDF.
  - Работа с файловой системой, обработка и отображение файлов различных форматов.
  - Поддержка протоколов LDAP, IMAP, SNMP, NNTP, POP3.
  - Огромное количество расширений и готовых проектов для любых вариантов Интернет-технологий

МЕСТО PHP

Рейтинг технологий stackoverflow.com



Индекс популярности TIOBE

Sep 2019	Sep 2018	Change	Programming Language
1	1		Java
2	2		C
3	3		Python
4	4		C++
5	6	↑	C#
6	5	↓	Visual Basic .NET
7	8	↑	JavaScript
8	9	↑	SQL
9	7	↓	PHP
10	10		Objective-C
11	34	⇈	Groovy
12	14	↑	Assembly language
13	11	↓	Delphi/Object Pascal

ПРЕИМУЩЕСТВА PHP

Главной особенностью PHP является его практичность, т.к. язык предоставляет программисту средства для быстрого и эффективного решения поставленных задач. Данная особенность PHP обусловлена пятью важными характеристиками:

- Простота
- Гибкость
- Традиционность
- Эффективность
- Безопасность

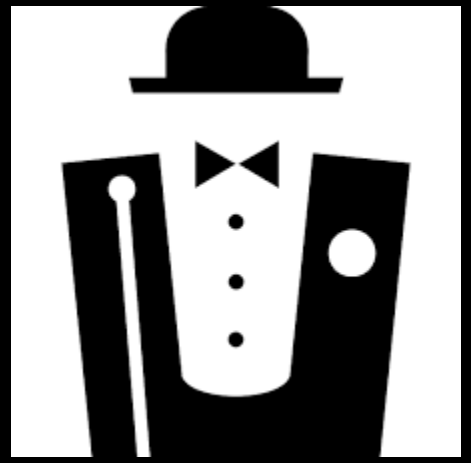
Расмус Лердорф (Rasmus Lerdorf) — датский программист, написавший в 1994 году набор скриптов, обрабатывающих шаблоны HTML-документов, позже воплотившийся в интерпретатор языка сценариев PHP, с помощью которого можно было решать различные задачи веб-приложений, включая создание сайтов для различных целей и направлений.

Актуальная на 2019 год стабильная версия – PHP 7.3

PHP ДЛЯ WEB-РАЗРАБОТКИ

Связка двух технологий — Apache и PHP — представляет собой наиболее удобное решение для небольших и средних по размеру сайтов. Обе программы разрабатываются на принципах OpenSource и совершенно бесплатны.

Денвер (Denwer) — русский проект, набор дистрибутивов и программная оболочка, предназначенные для создания и отладки сайтов (веб-приложений, динамического содержимого интернет-страниц) на локальном компьютере (локальный сервер, без необходимости подключения к сети Интернет) под управлением MS Windows.



- Инсталлятор (поддерживается также инсталляция на flash-накопитель).
- Apache, SSL, SSI, mod_rewrite, mod_php.
- PHP5 с поддержкой GD, MySQL, sqLite.
- MySQL5 с поддержкой транзакций.
- Система управления виртуальными хостами.
- Система управления запуском и завершением всех компонентов Денвера.
- phpMyAdmin — система управления MySQL через Web-интерфейс.
- Эмулятор sendmail и SMTP-сервера

ОСНОВЫ СИНТАКСИСА

Программа на php встраивается в HTML-документ

```
<?php
  echo "Это обычная инструкция для обработки PHP";
?>

<?
  echo "Простейшая инструкция";
?>

<script language="php">
  echo "Некоторые редакторы (FrontPage) предпочитают делать так";
</script>

<%
  echo "Можно использовать теги в стиле ASP (но не нужно) ";
  // однострочный комментарий
  # тоже вариант однострочного комментария
  /* многострочный
     комментарий */
%>
```

Расширение файла с PHP-кодом зависит от настроек web-сервера:

- example.php
- example.phtml

ОСНОВЫ СИНТАКСИСА

Переменные и константы

Переменная в PHP обозначается знаком доллара, за которым следует ее имя ($my_var). Правильное имя переменной должно начинаться с буквы или символа подчеркивания.

Имя переменной чувствительно к регистру.

```
<?php
  $first = ' Text ';
  $second = $first; // по ссылке
  echo "Значение second =" .$second;
  $zzz = & $first;
  $first = ' New text ‘;
  echo "Значение zzz =" .$zzz;
?>
```

Константы типа *boolean*: True, False
echo (int)TRUE даст в результате 1

```
<?php
  define("PASSWORD", "qwerty");
  // определяем константу PASSWORD
  define("PI","3.14", True);
  // регистронезависимая  константа PI
  echo (PASSWORD);
  // выведем значение константы: qwerty
  echo pi;
  // выведет 3.14, т.к. PI регистронезависима
?>
```

ОСНОВЫ СИНТАКСИСА

Операторы и операции

PHP – слаботипизированный язык

+	Сложение	$c = $a + $b
-	Вычитание	$c = $a - $b
*	Умножение	$c = $a * $b
/	Деление	$c = $a / $b
%	Остаток от деления	$c = $a % $b
.	Конкатенация (сложение строк)	$c = $a . $b
>=	Больше или равно	$c = $a >= $b

=	**$a = ($b = 4) +5;** ($a будет равна 9, $b будет равна 4)
+=	**$a += 5;** // эквивалентно $a = $a + 5;
.=	$b = "Привет "; **$b .= "всем";** (эквивалентно $b = $b . "всем";) В результате: b = "Привет всем"
&&	**$c = $a && $b;** // логическое «И»
||	**$c = $a || $b;** // логическое «ИЛИ»
==	**$a == $b** // условие равенства значений
===	**$a === $b** // условие равенства значений и типов
++	**++$a** // Пре-инкремент **$a++** // Пост-инкремент

ОСНОВЫ СИНТАКСИСА

Типы данных

Скалярные:
- boolean (логический)
- integer (целый)
- float (с плавающей точкой)
- string (строковый)

Смешанные:
- array (массив) ;
- object (объект).

Специальные:
- resource (ресурс)
- NULL

```
<?php
  $a = 1234;        //десятичное число
  $a = -123;        //отрицательное число
  $a = 0123;        //восьмеричное число
  $a = 0x1A;        //шестнадцатеричное число

  $b = 1.234; // по умолчанию «двойная точность»
  $b = 1.2e3;
  $b = 7E-10;

  $str1 = 'Строка 1 '; // ограничения на размер нет
  $str2 = "Строка 2 ";
  $str3 = <<<EOD
Третья строка - это $str1 и $str2
  EOD;
?>
```

ОСНОВЫ СИНТАКСИСА

Тип данных - массив

Определить массив можно либо с помощью конструкции array(), либо непосредственно задавая значения его элементам.

```
array (key1 => value1, key2 => value2, ... )
```

т.н. ассоциативный массив

```
<?php
  $books = array ("php" =>  "PHP users guide", 12 => True);
  echo $books["php"]; //выведет "PHP users guide"
  echo $books[12];  //выведет 1
?>
```

```
<?php
  $books[12]= 333;
  $books[] = "test"; // ключ будет равен 13
  unset($books[12]); //удаляет элемент № 12
  unset($books); //удаляет массив полностью
?>
```

```
<?php
// обычный вариант одномерного массива
  $A[0]="Апельсин";
  $A[1]="Банан";
  $A[2]="Груша";
  $A[3]="Помидор";
  echo $A[3]
?>
```

```
<?php
// Многомерный ассоциативный массив
  $B["Ivanov"] = array("name"=>"Иванов И.И.", "age"=>"25“;
  $B["Petrov"] = array("name"=>"Петров П.П.", "age"=>"34“;
  $B["Sidorov"] = array("name"=>"Сидоров С.С.", "age"=>"47";
?>
```

ОСНОВЫ СИНТАКСИСА

Операции с массивами

Сложение массивов

```
<?
  $a = array("и"=>"Информатика", "м"=>"Математика");
  $b = array("и"=>"История", "м"=>“Магия", "ф"=>"Физика");
  $c = $a + $b;
  $d = $b + $a;
  print_r($c); // получим: Array([и]=>Информатика  [м]=>Математика [ф]=>Физика)
  print_r($d); // получим: Array([и]=>История  [м]=>Магия [ф]=>Физика)
?>
```

Равенство *массивов* (==) – это когда совпадают все пары *ключ / значение* элементов массивов. Эквивалентность (===) – когда кроме равенства *значений* и *ключей* элементов требуется, чтобы элементы в обоих массивах были записаны в одном и том же порядке.

```
<?php
  // Простой способ инициализации массива
  $names[]="Апельсин";
  $names[]="Банан";
  $names[]="Груша";
  $names[]="Помидор";
?>
```

ОСНОВЫ СИНТАКСИСА

Тип данных - объект

Класс – это набор объектов, обладающих определенными свойствами и методами работы с ним, а объект – экземпляр соответствующего класса.

```
<?php

class Person // Создаем класс
{
  function info()
    //Объявляем метод данного класса
    {  echo “Это PHP-программист"; }
}

$bob = new Person;  // Создаем объект класса
$bob -> info(); // Вызываем метод класса
}

?>
```

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

```
Объект = new Имя_класса;
```

```
function Getname() {
  echo $this->name;
}

function
Setname($name) {
  $this->name = $name;
}
```

ОСНОВЫ СИНТАКСИСА

Смешанные типы данных

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.).

Специальное значение NULL говорит о том, что переменная не имеет значения.
Переменная считается NULL, если

- ей была присвоена константа NULL ($var = NULL);
- ей ещё не было присвоено какого-либо значения;
- она была удалена с помощью unset().

УПРАВЛЯЮЩИЕ КОНСТРУКЦИИ

Условные операторы

Оператор **if**

```
<?
 $names = array("Иван", "Петр", "Семен");
 if ($names[0] == "Иван")
 {
    echo "Привет, Ваня!";
   $num = 1;    $account = 2000;
 }
 if ($num) echo "Появился первый в списке!";
?>
```

```
<?
if ($a > $b) {
    echo "a больше, чем b";
} elseif ($a == $b) {
    echo "a равен b";
} else {
    echo "a меньше, чем b";
}
?>
```

```
<?php
 $a = 10;
 $b = 5;
 $c = 2;
 if ($a + $b > $c and $c < $a) {
   echo 'Истина';
 }
?>
```

```
<?php
 $my = True;
 echo $my ? 'Переменная истина' : 'Ложь';
?>
```

УПРАВЛЯЮЩИЕ КОНСТРУКЦИИ

Условные операторы

Оператор **switch**

```
switch (анализируемое_выражение) {
  case значение1:    блок_действий1 break;
  case значение2:    блок_действий2 break;
  ...
  default:    блок_действий_по_умолчанию
}
```

```
<?
 $names = array("Иван", "Петр");
 switch ($names[0]) {
   case "Иван":  echo "Привет, бро!";
   break;
   case "Петр":    echo  "Наше почтение!";
   break;
   default:  echo "Не знаем, кто вы, $names[0], но всё равно здравствуйте?";
 }
?>
```

```
<?php
switch ( $i ) {
    case 0:
    case 1:
    case 2:
        echo "i меньше чем 3, но неотрицательный";
        break;
    case 3:
        echo "i равно 3";
}
?>
```

УПРАВЛЯЮЩИЕ КОНСТРУКЦИИ

Циклы **while**

```
<?
 $i = 1;
 while ($i < 10) {
  if ($i % 2 == 0) print $i;
  $i++;
 }
?>
```

Значение выражения проверяется каждый раз в начале цикла, так что, даже если его значение изменилось в процессе выполнения блока, цикл не будет остановлен до конца итерации

```
<?
 $i = 1;
 do {
  if ($i % 2 == 0) print $i;
  $i++;
 } while ($i < 10)
?>
```

Истинность выражения проверяется в конце цикла. Благодаря этому блок цикла do...while гарантированно выполняется хотя бы один раз.

```
<?php
$x = 0;
while ($x++ < 10) echo $x;
// Выводит 12345678910
?>
```

УПРАВЛЯЮЩИЕ КОНСТРУКЦИИ

Циклы

Цикл **for**

for (выражение1; выражение2; выражение3) { блок_выполнения }

Выражение1 вычисляется безусловно один раз в начале цикла. В начале каждой итерации вычисляется выражение2. Если оно является True, то цикл продолжается и выполняются все команды блока_выполнения. Если выражение2 вычисляется как False, то исполнение цикла останавливается. В конце каждой итерации (т.е. после выполнения всех команд блока_выполнения) вычисляется выражение3.

```
<?php
  for ($i = 0; $i < 10; $i++) {
    if ($i % 2 == 0) print $i;
  }
?>
```

```
<?php
  for ($i=0, $j=0, $k="Точки"; $i<10; $j++, $i+=$j) {
    $k = $k."."; echo $k;
  }
  // Выводит Точки.Точки..Точки...Точки....
?>
```

УПРАВЛЯЮЩИЕ КОНСТРУКЦИИ

Циклы

Цикл **foreach**

foreach ($array as $value) {блок_выполнения}
foreach ($array as $key => $value) {блок_выполнения}

В первом случае формируется цикл по всем элементам массива, заданного переменной $array. На каждом шаге цикла значение текущего элемента массива записывается в переменную $value, и внутренний счетчик массива передвигается на единицу. Выполнение блока_выполнения происходит столько раз, сколько элементов в массиве $array.

Во второй форме записи на каждом шаге цикла ключ текущего элемента массива записывается в переменную $key, которую можно использовать в блоке_выполнения.

```
<?php
 $names = array("Иван", "Петр", "Семен");
 foreach ($names as $val) {
   echo "Привет, $val <br>";
 }
?>
```

```
<?php
 $a = array(
   "one" => 1,
   "two" => 2,
   "three" => 3,
   "seventeen" => 17
 );

 foreach ($a as $k => $v) {
   echo "\$a[$k] => $v.\n";
 }
?>
```

ФУНКЦИИ В PHP

В языке программирования PHP существует 2 вида функций:

1. Функции, определяемые пользователем
2. Внутренние (встроенные) функции

```
function Имя_функции (параметр1, параметр2,  ... параметрN) {
   Блок_действий
   return возвращаемое_значение;
}
```

С помощью аргументов данные в функцию можно передавать тремя различными способами:

- по значению
- по ссылке
- значения по умолчанию

```
<?php
// рекурсивная функция вычисления факториала
  function fact($n) {
    if ($n==0) return 1;
    else return $fact = $n * fact($n-1);
  }

  echo 'Факториал 5 равен '.fact(5).'<br>';
  echo 'Факториал 50 равен '.fact(50).'<br>';
?>
```

ФУНКЦИИ В PHP

Примеры

```php
<?php
 function sayHello($Name){
   echo "Привет, ". $Name . "!<br>";
 }
?>
```

```php
<?php
 function makecoffee( $type = "капучино" )
 {
    return "Готовим чашку $type.\n";
 }

 echo makecoffee(); // по умолчанию
 echo makecoffee(null);
 echo makecoffee("эспрессо");
?>
```

```php
<?php
 function mySum($numX, $numY) {
   $total = $numX + $numY;
   return $total;
 }

 $myNumber = 0;
 $myNumber = mySum(3, 4);
 echo "myNumber = " . $myNumber ."<br>";
?>
```

```php
<?php
 function fuf( &$string )
 {
    $string .= 'а эта - внутри.';
 }

 $str = 'Эта строка за пределами функции, ';
 fuf( $str );
 echo $str;
?>
```

ПЕРЕМЕННЫЕ ВНУТРИ ФУНКЦИИ

Чтобы использовать внутри функции переменные, заданные вне её тела, эти переменные нужно объявить как глобальные. Для этого в теле функции следует перечислить их имена после ключевого слова global: global $var1, $var2;

Чтобы использовать переменные только внутри функции, при этом сохраняя их значения и после выхода из функции, нужно объявить эти переменные как статические. Объявление таких переменных производится с помощью ключевого слова static: static $var1, $var2;

```
<?php
 function Test(){
   global $a;
   static $b = 2;
   $a = $a * $b;
   $b++;
 }

 $a=1;
 echo 'Изначально $a='.$a.'<br>';
 Test();
 echo 'Теперь $a='.$a.'<br>';
 Test();
 echo 'И, наконец, $a='.$a.'<br>';
?>
```

ВСТРОЕННЫЕ ФУНКЦИИ

- В PHP содержится достаточно большое количество встроенных функций и языковых конструкций. Также есть функции, которые требуют, чтобы PHP был собран с определенными расширениями, в противном случае будут генерироваться фатальные ошибки, вызванные использованием неизвестной функции.
- Например, для того чтобы использовать функции для работы с изображениями, например, imagecreatetruecolor(), необходимо собрать PHP с поддержкой GD.
- Или же для того, чтобы воспользоваться функцией mysqli_connect(), необходима поддержка модуля MySQLi. Тем не менее, есть много встроенных функций, которые доступны всегда: например функции обработки строк и математические функции.
- Вызвав специальные функции phpinfo() или get_loaded_extensions(), можно узнать, поддержка каких модулей есть в используемом PHP.

ВСТРОЕННЫЕ ФУНКЦИИ

Функции для работы с переменными

- empty — Проверяет, пуста ли переменная
- gettype — Возвращает тип переменной
- is_array — Определяет, является ли переменная массивом
- is_bool — Проверяет, является ли переменная булевой
- is_int — Проверяет, является ли переменная целым числом
- is_null — Проверяет, является ли значение переменной равным NULL
- isset — Определяет, была ли установлена переменная значением, отличным от NULL
- unset — Удаляет переменную
- var_dump — Выводит информацию о переменной

ВСТРОЕННЫЕ ФУНКЦИИ

Примеры функций для работы с массивами

```
<?
  $del_items = array("langs" => array(
    "10"=>"Python", "12"=>"Lisp"),
    "other"=>"Информатика");

  echo count($del_items) . "<br>";
    // выведет 2
  echo count($del_items, COUNT_RECURSIVE);
    // выведет 4
?>
```

Пример использования функции count для вычисления количества элементов массива. Фактически элементов массива 2, а именно пары ключ-значение: "langs" => array() и "other"=>"Информатика". Однако, если учесть, что ключ "langs" содержит массив из двух элементов, то общее количество элементов будет равно 4.

```
<?
 $items = array(10 => "хлеб", 20 => "молоко",
 30 => "бутерброд");
 sort($items);
 // [0] => бутерброд [1] => молоко [2] => хлеб
 print_r($items);
?>
```

Функция sort упорядочивает значения массива по возрастанию. Эта функция удаляет все существовавшие в массиве ключи, заменяя их числовыми индексами, соответствующими новому порядку элементов. В случае успешного завершения работы она возвращает true, иначе – false.

ВСТРОЕННЫЕ ФУНКЦИИ

Примеры функций для работы со строками

Для того чтобы определить, входит ли подстрока в состав строки, используется функция strpos()

```
<?
$str = "Идея наносить данные на перфокарты
и затем считывать и обрабатывать их
автоматически принадлежала Джону Биллингсу,
а ее техническое решение осуществил Герман
Холлерит. Перфокарта Холлерита оказалась
настолько удачной, что без малейших изменений
просуществовала до наших дней.";

  $pos = strpos($str,"Холлерит");
  if ($pos !== false) echo "Искомая строка
   встречена в позиции номер $pos ";
  else echo "Искомая строка не найдена";
?>
```

Подстроку из строки можно выделить с помощью функции substr()

```
<?php
$word = “say <b>Hello, world!</b><br>";
echo $word
$pure_str = substr($word, 7, -8);
echo $pure_str;
?>
```

str_replace — Заменяет все вхождения строки поиска на строку замены

substr_replace — функция заменяет часть строки строкой, предназначенной для замены

РЕЗЮМЕ

Язык PHP — это удобный инструмент, который всё ещё продолжает развиваться, имеет большую поддержку в Интернет-сообществе, огромное количество готовых библиотек и решений. Немаловажно, что он прекрасно документирован на разных языках.

Идея работы с PHP — обладая нулевым уровнем подготовки быстро стартовать, без проблем сформировать хорошую команду и сделать продукт «не хуже чем у других».

Язык PHP имеет массу проблем, в том числе и на уровне идеологии. Считать его современным перспективным языком сейчас никто не станет. Но и игнорировать его существование пока ещё рано.


YOUR ATTENTION
I THANK YOU FOR