О возможностях PHP и встроенных функциях для ежедневного использования

Заметка: активирована адаптивная версия сайта, которая автоматически подстраивается под небольшой размер Вашего браузера и скрывает некоторые детали сайта для удобства чтения. Приятного просмотра!

23.09.2014 **Раздел:** Создание сайта / Учебник по PHP 5

Всем доброго дня! Продолжаем изучать наш любимый PHP. Сегодня вы узнаете об основных функциях PHP, которые понадобятся вам в первую очередь.

Возможности языка

Позвольте мне сделать краткое лирическое отступления и рассказать, что на PHP можно писать не только сайты. Его можно использовать и для создания оконных (десктоп) приложений или консольных приложений, которые будут работать как и любое другое приложение Windows, пользователям не понадобиться для этого браузер, а вам не понадобиться сервер. Конечно, PHP является далеко не лучшим выбором для создания десктоп приложений, однако такая возможность есть.

Кроме этого PHP, помимо создания web-страничек (HTML), может генерировать настоящие PDF файлы, XML, изображения, диаграммы и даже Flash ролики. Вдобавок к сохранению всего вышеперечисленного на жёсткий диск, PHP может генерировать и отдавать файлы пользователям "на лету", без сохранения их куда-либо.

В прошлых статьях мы узнали, что такое функции, зачем они нужны и как

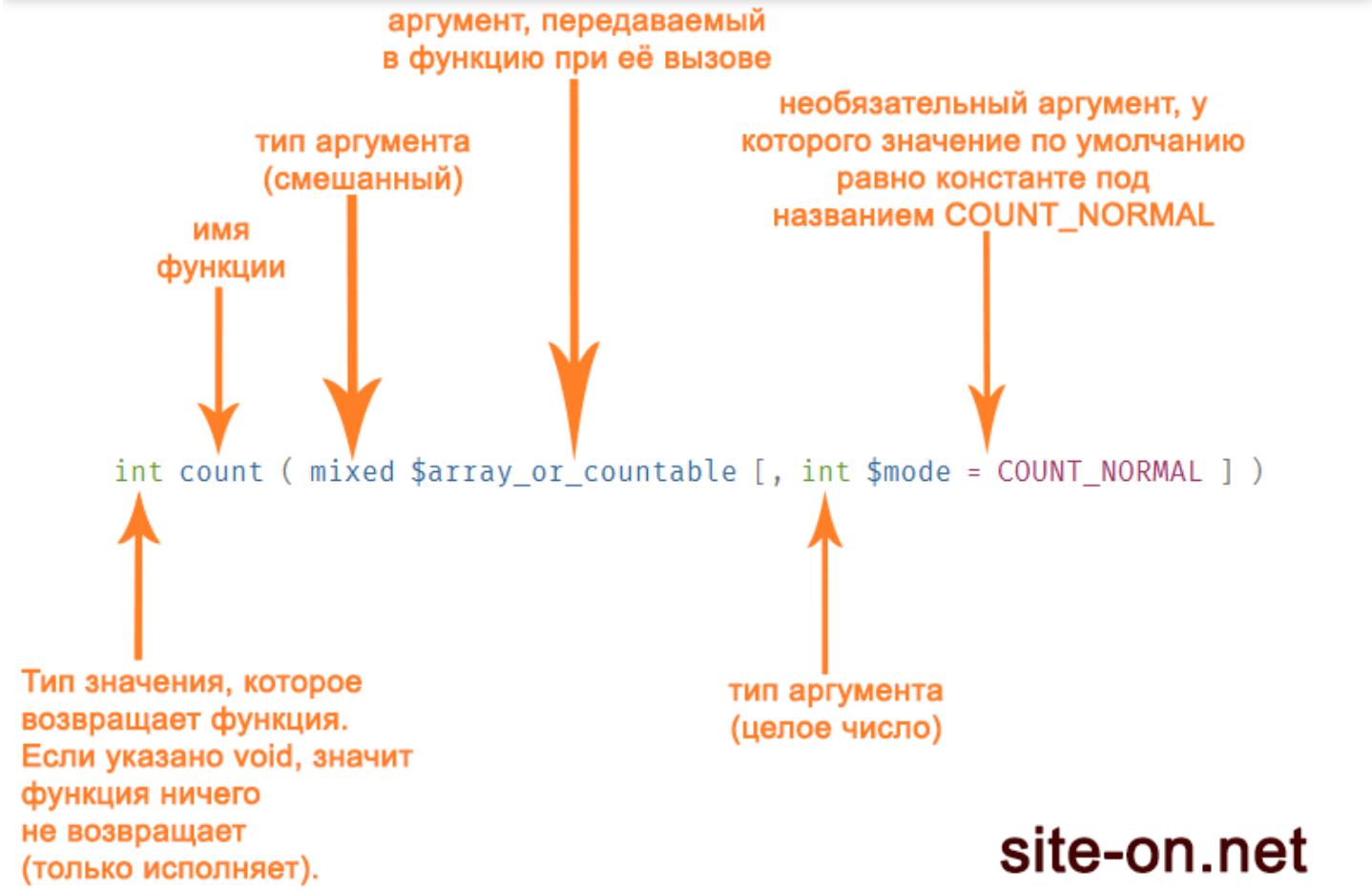
функции, ведь в PHP уже есть сотни встроенных функций, которые станут вашими незаменимыми и надёжными помощниками при разработке веб-приложений.

Так как среди моих читателей многие делают сайты на популярных CMS (Joomla, Wordpress и других), а также многие из них мало знакомы с PHP и программированием в целом, то спешу объяснить, что встроенные функции PHP можно использовать в любой CMS, в любом месте! И при этом от вас не требуется ничего настраивать и устанавливать дополнительно, так как встроенные функции, в отличие от пользовательских, вшиты в ядро PHP и никуда от туда деться не могут.

Список всех встроенных функций и возможностей PHP в целом можно узнать на официальном сайте: **http://php.net** в разделе документации. Но прежде чем вы ринетесь изучать документацию или продолжите чтение данной статьи, я должен объяснить вам, как пользоваться документацией, причём документацией не только по PHP, но и любого другого языка программирования.

Как пользоваться документацией по PHP

Предлагаю рассмотреть всё на примере упомянутой в прошлой статье функции count():



Как комментарий к картинке хочу отметить, что тип аргумента **mixed** – означает, что можно передавать несколько различных **типов**, но это не значит, что разрешено передавать любой тип. Какие именно типы можно передавать в каждую конкретную функцию читайте в более подробном описании к ней.

Что означает та или иная константа, установленная как значение по умолчанию, также нужно читать в полном описании к функции.

А теперь, встречайте список функций, которые вам обязательно пригодятся!

Основные функции PHP

Функции для работы с типами

gettype() - Возвращает тип переменной. Возможными значениями возвращаемой строки являются:

- "boolean"
- "integer"

- "double" (по историческим причинам в случае типа float возвращается "double")
- "string"
- "array"
- "object"
- "resource"
- "NULL"
- "unknown type" - неизвестный тип

Вообще функцию **gettype** используют вместе с **echo**, для вывода типа переменной на экран. Если вам нужно проверить тип и в зависимости от результата пустить код по той или иной ветке (с помощью **оператора if**), используйте функции **is_**, о них дальше.

settype() - Присваивает переменной новый тип.

```
$foo="5bar"; // строка
settype($foo,"integer"); // $foo теперь 5 (целое число, то есть integer)
```

Допустимыми значениями типа являются:

- "boolean" (или "bool")
- "integer" (или "int")
- "float" (или "double")
- "string"
- "array"
- "object"
- "null"

Функции **is_array()**, **is_bool()**, **is_float()**, **is_int()**, **is_null()**, **is_string()**, **is_numeric()** - определяют, является ли переменная тем или иным типом.

Приведение типов

```
$foo = 10; // $foo это целое число
$bar = (boolean)$foo; // $bar это булев тип
```

- (int), (integer) - приведение к целому числу
- (bool), (boolean) - приведение к true или false
- (float), (double), (real) - приведение к числу с плавающей точкой (например: 56.83452)
- (string) - приведение к строке
- (array) - приведение к массиву
- (object) - приведение к объекту
- (unset) - приведение к NULL (появилось в PHP 5)

Проверка на существование

Следующие 2 функции: **isset()** и **empty()** - лично я очень люблю и использую очень часто.

```
if (isset($var)){
    /* Отвечает, существует (определена) ли переменная. Возвращает TRUE,
    если переменная существует или FALSE в противном случае.*/
}

if (empty($var)){
    /* Определяет, считается ли переменная пустой. */
}
```

Переменная считается пустой, если она:

- "" или '' (пустая строка)
- 0 (целое число)
- 0.0 (дробное число)
- "0" (строка)
- NULL
- FALSE
- array() (пустой массив)
- var $var; (переменная объявлена, но не имеет значения в классе)

Возвращает TRUE, если переменная пустая или FALSE, если переменная не пустая.

Сравнение типов и значений, возвращаемых функциями, о которых мы говорили выше:

Выражение	gettype()	empty()	is_null()	isset()	boolean : if($x)
$x = "";	string	TRUE	FALSE	TRUE	FALSE
$x = null	NULL	TRUE	TRUE	FALSE	FALSE
var $x;	NULL	TRUE	TRUE	FALSE	FALSE
$x неопределена	NULL	TRUE	TRUE	FALSE	FALSE
$x = array();	array	TRUE	FALSE	TRUE	FALSE
$x = false;	boolean	TRUE	FALSE	TRUE	FALSE
$x = true;	boolean	FALSE	FALSE	TRUE	TRUE
$x = 1;	integer	FALSE	FALSE	TRUE	TRUE
$x = 42;	integer	FALSE	FALSE	TRUE	TRUE
$x = 0;	integer	TRUE	FALSE	TRUE	FALSE
$x = -1;	integer	FALSE	FALSE	TRUE	TRUE
$x = "1";	string	FALSE	FALSE	TRUE	TRUE
$x = "0";	string	TRUE	FALSE	TRUE	FALSE
$x = "-1";	string	FALSE	FALSE	TRUE	TRUE
$x = "php";	string	FALSE	FALSE	TRUE	TRUE
$x = "true";	string	FALSE	FALSE	TRUE	TRUE
$x = "false";	string	FALSE	FALSE	TRUE	TRUE

Для работы с массивами

count() - Считает количество элементов массива или количество свойств объекта.

```
$result = count($array);
```

print_r() - Выводит удобочитаемую информацию о переменной (в том числе о массивах и объектах):

```
<pre><?php print_r($array);?></pre>
```

var_dump() - Выводит более полную информацию о переменной (в том числе о массивах и объектах), по сравнению с **print_r**.

```
<pre><?php var_dump($array);?></pre>
```

Функции для манипулирования указателем массива:

```
$array = array('первый', 'второй', 'третий', 'четвертый');
// По умолчанию внутренний указатель массива указывает на первый элемент:
echo current($array); // вернёт строку "первый"

// Передвигает внутренний указатель массива на одну позицию вперёд:
echo next($array); // вернёт строку "второй"
echo current($array); // вернёт строку "второй"

// Устанавливаем внутренний указатель массива на его последний элемент:
echo end($array); // вернёт "четвертый"

// Передвигаем внутренний указатель массива на одну позицию назад:
echo prev($array); // вернёт "третий"

// Устанавливаем внутренний указатель массива на его первый элемент:
echo reset($array); // вернёт "первый"

// Получаем ключ текущего элемента
echo key($array); // вернёт 0
```

Хочу обратить ваше внимание на функцию **array_shift()**, её можно использовать как просто для удаления первого элемента массива, так и для его сохранения в переменную, перед удалением. При этом все числовые ключи будут изменены таким образом, что нумерация массива начнётся с нуля, в то время как строковые ключи останутся прежними, благодаря чему пользоваться этой функцией очень удобно.

Если вам нужно удалить или извлечь с удалением последний элемент массива, используйте функцию **array_pop()**. Её работа полность аналогична работе функции **array_shift()**.

array_key_exists() - Проверяет, присутствует ли в массиве указанный ключ или индекс.

in_array() - Проверяет, присутствует ли в массиве значение.

array_keys() - Возвращает все или некоторое подмножество ключей массива.

array_values() - Выбирает все значения массива.

array_merge() - Сливает один или большее количество массивов в один

array_diff() - Вычисляет расхождение массивов.

array_intersect() - Вычисляет схождение массивов.

Сортировка массивов

sort() - Эта функция сортирует массив. После завершения работы функции элементы массива будут расположены в порядке возрастания.

ksort() - Сортирует массив по ключам, сохраняя отношения между ключами и значениями. Эта функция полезна, в основном, для работы с ассоциативными массивами.

asort() - Эта функция сортирует массив таким образом, что сохраняются отношения между ключами и значениями. Она полезна, в основном, при сортировке ассоциативных массивов, когда важно сохранить отношение ключ => значение.

rsort() – Сортирует массив в обратном порядке.

arsort() – Сортирует массив в обратном порядке, сохраняя ключи.

krsort() – Сортирует массив по ключам в обратном порядке.

natsort() – Сортирует массив, используя алгоритм "natural order".

natcasesort() – Сортирует массив, используя алгоритм "natural order" без учета регистра символов.

usort() – Сортирует массив по значениям используя пользовательскую функцию для сравнения элементов.

uksort() – Сортирует массив по ключам, используя пользовательскую функцию для сравнения ключей.

uasort() – Сортирует массив, используя пользовательскую функцию для сравнения элементов с сохранением ключей.

Математические функции

abs() - Модуль числа.

max() - Находит наибольшее значение.

min() - Находит наименьшее значение.

rand() - Генерирует случайное число.

round() - Округляет число типа float.

ceil() - Округляет дробь в большую сторону.

floor() - Округляет дробь в меньшую сторону.

Функции обработки строк

substr_count() - Возвращает число вхождений подстроки (указанной строки).

rtrim, ltrim, **trim()** - Удаление пробелов (или других символов) из начала и (или) конца строки.

strtolower, **strtoupper()** - Преобразование строки в нижний или верхний регистр.

lcfirst, **ucfirst()** - Преобразование первого символа строки в нижний или верхний регистр.

substr() - Возвращает подстроку (часть строки).

number_format() - Возвращает отформатированное число, можно получить различный вид числа, например: 123,234.56

str_replace() - Заменяет все вхождения строки поиска на строку замены.

explode() – Разбивает строку на подстроки в массив. В роли разделителя выступает заданный символ или строка. Каждая разбитая подстрока (часть строки) – это ячейка массива.

implode() - противоположность **explode()**. Функция склеивает массив в одну строку.

Функции для работы с датой и временем

getdate() - Возвращает ассоциативный массив (array), содержащий информацию о дате, представленной меткой времени timestamp, или текущем системном времени, если timestamp не был передан. Пример:

```
$today = getdate();
print_r($today);
//Результат:
[seconds] => 40
[minutes] => 58
[hours] => 21
[mday] => 17
[wday] => 2
[mon] => 6
[year] => 2003
[yday] => 167
[weekday] => Tuesday
[month] => June
[0] => 1055901520
```

time() - Возвращает количество секунд, прошедших с начала Эпохи Unix (The Unix Epoch, 1 января 1970 00:00:00 GMT) до текущего времени. Результатом будет что-то типа 1234567890.

date() - Возвращает строку со временем, отформатированную в соответствии с указанным форматом, используя метку времени, заданную аргументом timestamp или текущее системное время, если timestamp не задан.

strtotime() - Преобразует текстовое представление даты на английском языке в метку времени Unix. Первым параметром функции должна быть строка с датой на английском языке, которая будет преобразована в метку времени относительно метки времени, переданной в now, или текущего времени, если аргумент now опущен. Примеры использования:

```
$dt = strtotime("now");
$dt = strtotime("10 September 2000");
$dt = strtotime("+1 day");
$dt = strtotime("+1 week");
$dt = strtotime("+1 week 2 days 4 hours");
$dt = strtotime("next Thursday");
$dt = strtotime("last Monday");
```

Заключение

Конечно же, при разработке используется ещё большое множество других встроенных функций PHP, однако функции, которые я перечислил выше, скорее всего, понадобятся вам в первую очередь. Учить наизусть их не нужно, вполне достаточно просто знать об их существовании и при необходимости уметь быстро найти информацию о том, как их использовать. В ходе постоянной практики вы автоматически их запомните.

<< Предыдущая статья Следующая статья >>

Другие статьи по теме функций в PHP

1. Как автоматически вставить рекламу в статьи Joomla и Wordpress
2. Собственные (пользовательские) функции в PHP. Часть 1
3. Все нюансы работы с функциями в PHP. Часть 2
4. Красивая, удобная и компактная пагинация для Joomla 2.5

Пожалуйста, оцените эту статью

Применить

Средняя оценка: 4.61 из 5 (проголосовало: 31)

Статья оказалась вам полезной? Подпишитесь, чтобы не пропустить новые!

Ваш email: Подписаться!

Вы можете помочь развитию проекта, сделав всего 1 клик:

Спасибо!

Пожалуйста, прокомментируйте, как Вам моя статья?

Имя:

Комментарий:

Если Вы хотите вставить код, пожалуйста, заключайте его в [code][/code]

Подписаться на новые комментарии:

Защита от спама: у треугольника три...

Ответ:

Подписаться на новые комментарии без комментирования - Email:

Защита от спама: у треугольника три...

Ответ:

28.06.2021 17:48:32 **ржака**:
иди нахуй рваножопый пидор!

Ответить на комментарий