

ТОП рейтинги

Самые интересные рейтинги различных тематик

Php топ 100 функций

РЕКЛАМА

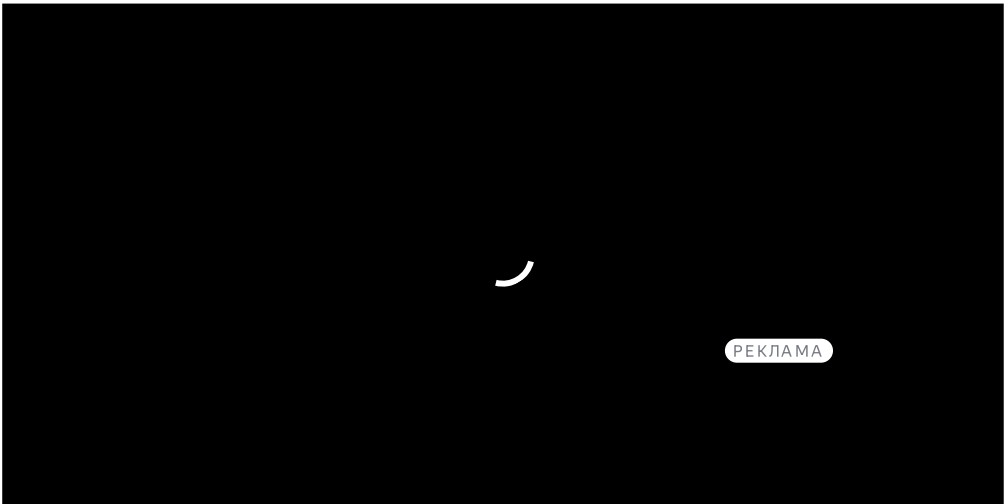
Покупайте на [hoff.ru](#)

***Рекламодатель:** ООО «Домашний Интерьер», 123290, Москва, 1-й Магистральный пр., д. 11, стр. 1, пом. II, этаж 2, ком. 54, ОГРН 1077763747269, ИНН 7709770002. Подробнее на Hoff.ru.

РЕКЛАМА

Покупайте на [hoff.ru](#)

***Рекламодатель:** ООО «Домашний Интерьер», 123290, Москва, 1-й Магистральный пр., д. 11, стр. 1, пом. II, этаж 2, ком. 54, ОГРН 1077763747269, ИНН 7709770002. Подробнее на Hoff.ru.



ТОП 16 PHP функций, что я использую каждый день

// 30 марта, 2013 | 3788 просмотров

Решил написать ТОП 16 PHP функций, что я использую каждый день.

Будет полезно как начинающим разработчикам, так и профессионалам (хотя у них этот список может отличаться).

Это MUST HAVE функции, что обязательно нужно знать и уметь ими оперировать.

`int strlen (string $string)` Возвращает длину строки *string*. `bool empty (mixed $var)` Проверяет, пуста ли переменная. `string trim (string $str [, string $charlist])` Удаляет пробелы (или другие символы) из начала и конца строки

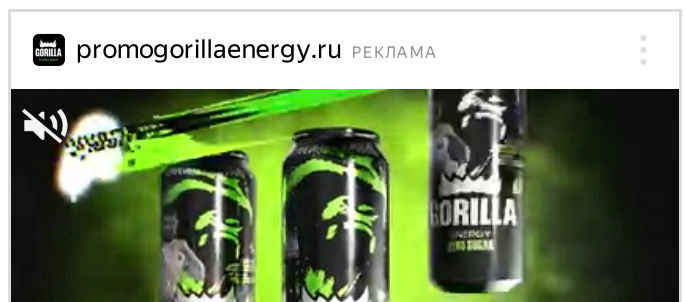
Эта функция возвращает строку *str* с удаленными из начала и конца строки пробелами.

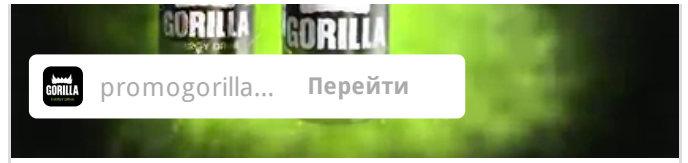
`int intval (mixed $var [, int $base = 10])` Возвращает целое значение переменной *var*, используя указанное основание системы исчисления *base* для преобразования (по умолчанию основание равно 10). `intval()` нельзя использовать с объектами, попытка это сделать вызовет ошибку уровня *E_NOTICE* и вернет значение 1. `mixed preg_replace (mixed $pattern , mixed $replacement , mixed $subject [, int $limit = -1 [, int &$count]])` Выполняет поиск совпадений в строке *subject* с шаблоном *pattern* и заменяет их на *replacement*. `mixed str_ireplace (mixed $search , mixed $replace , mixed $subject [, int &$count])` Регистро-независимый вариант функции `str_replace()`.

Эта функция возвращает строку или массив, в котором все вхождения *search* в *subject* заменены на *replace* (без учета регистра символов). Если не нужны сложные правила поиска/замены, использование этой функции предпочтительнее `preg_replace()` с модификатором *i*. `string strip_tags (string $str [, string $allowable_tags])` Удаляет HTML и PHP-теги из строки

Эта функция пытается вернуть строку *str*, из которой удалены все NUL-байты, HTML и PHP теги. Для удаления тегов используется тот же автомат, что и в функции `fgetss()`.

`string htmlspecialchars (string $string [, int $flags = ENT_COMPAT | ENT_HTML401 [, string $encoding = 'UTF-8' [, bool $double_encode = true]]])` Преобразует специальные символы в HTML-сущности `mixed filter_var (mixed $variable [, int $filter = FILTER_DEFAULT [, mixed $options]])` Фильтрует переменную с помощью определенного фильтра.





- Фильтры валидации данных
- Очищающие фильтры
- Остальные фильтры
- Флаги, используемые в фильтрах

string **date** (string *\$format* [, int *\$timestamp* = *time()*]) Форматирует вывод системной даты/времени.

Возвращает строку, отформатированную в соответствии с указанным шаблоном *format*. Используется метка времени, заданная аргументом *timestamp*, или текущее системное время, если *timestamp* не задан. Таким образом, *timestamp* является необязательным и по умолчанию равен значению, возвращаемому функцией *time()*. int **strtotime** (string *\$time* [, int *\$now* = *time()*]) Преобразует текстовое представление даты на английском языке в метку времени Unix.

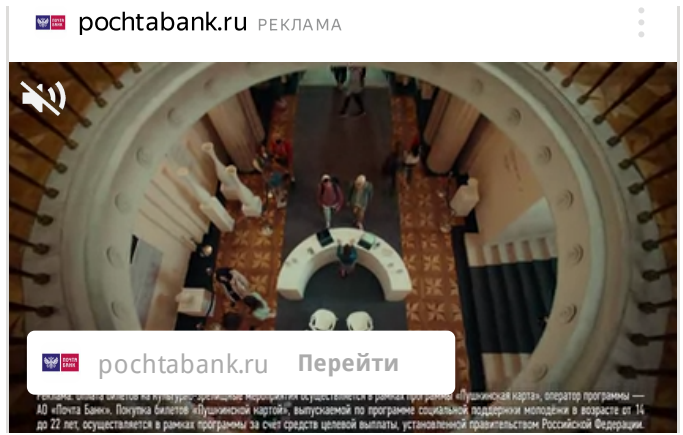
Первым параметром функции должна быть строка с датой на английском языке, которая будет преобразована в метку времени Unix (количество секунд, прошедших с 1 января 00:00:00 UTC) относительно метки времени, переданной в *now*, или текущего времени, если аргумент *now* опущен. string **join** (string *\$glue* , array *\$pieces*) Эта функция является псевдонимом: *implode()*.

Объединяет элементы массива с помощью строки *glue*. array **explode** (string *\$delimiter* , string *\$string* [, int *\$limit*]) Разбивает строку с помощью разделителя.

Возвращает массив строк, полученных разбиением строки *string* с использованием *delimiter* в качестве разделителя. array **array_РЕКЛАМА** (array *\$array1* [, array *\$...*]) : Сливает один или большее количество массивов.

Сливает элементы одного или большего количества массивов таким образом, что значения одного массива присоединяются к концу предыдущего. Результатом работы функции является новый массив. array **array_unique** (array *\$array* [, int *\$sort_flags* = *SORT_STRING*]) Убирает повторяющиеся значения из массива.

Принимает входной *array* и возвращает новый массив без повторяющихся значений. string **json_encode** (mixed *\$value* [, int *\$options* = 0]) Возвращает строку, содержащую JSON-представление *value*.



Источник

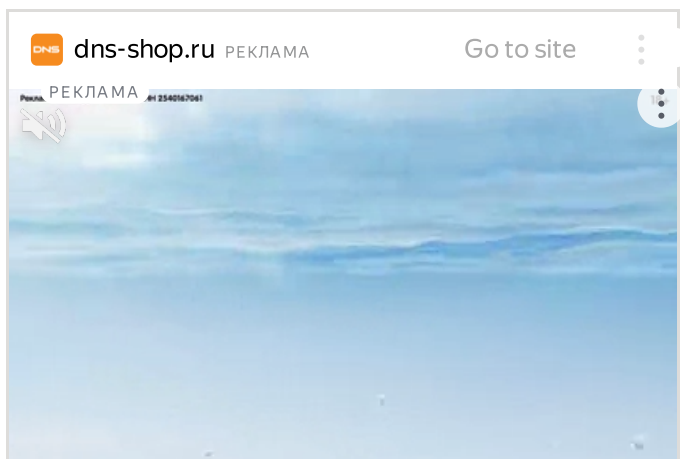
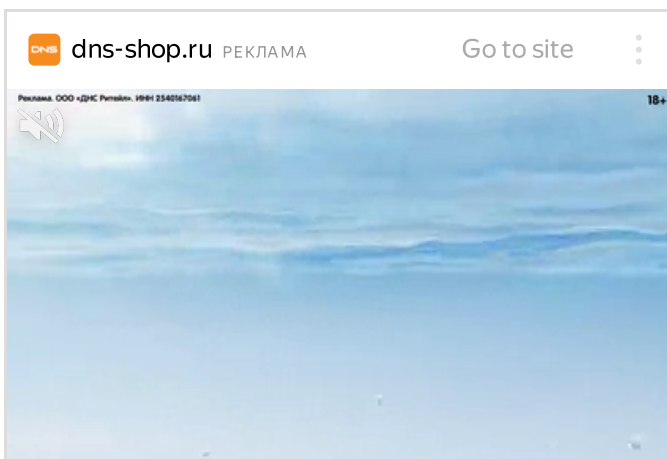
Читайте также: [Ручной запайщик пакетов рейтинг](#)

9 PHP функций, которые нужно знать всем

Сколько бы мы не использовали PHP, всё равно всплывают некоторые функции, о которых мы даже не слышали. Некоторые из них были бы нам очень полезны. Я создал небольшой список полезных функций, которые должны быть в арсенале каждого PHP программиста.

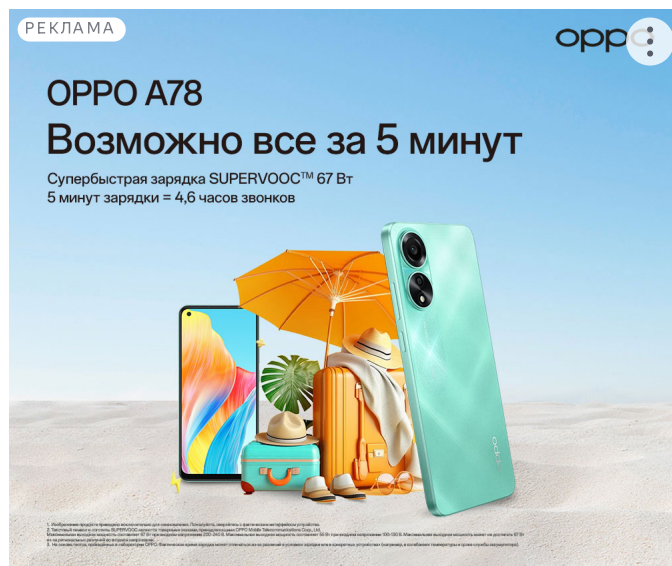
1. Создание функций с переменным числом аргументов

Скорее всего, вы уже знаете, что PHP позволяет нам создавать функции с необязательными аргументами. Сейчас я покажу функцию, в которой число аргументов может меняться от случая к случаю.



Но для начала, вспомним как мы создаём функции обычным образом:

Теперь посмотрим на то, как можно написать функцию с неограниченным количеством аргументов. Для этого будет использовать метод `func_get_args()`:



2. Используем `glob()` для поиска файлов

Часто названия функций говорят сами за себя. Такого нельзя сказать о функции `glob()`.

Если не вдаваться в подробности, её функциональность схожа с методом `scandir()`. Она позволяет найти необходимый файл по шаблону:

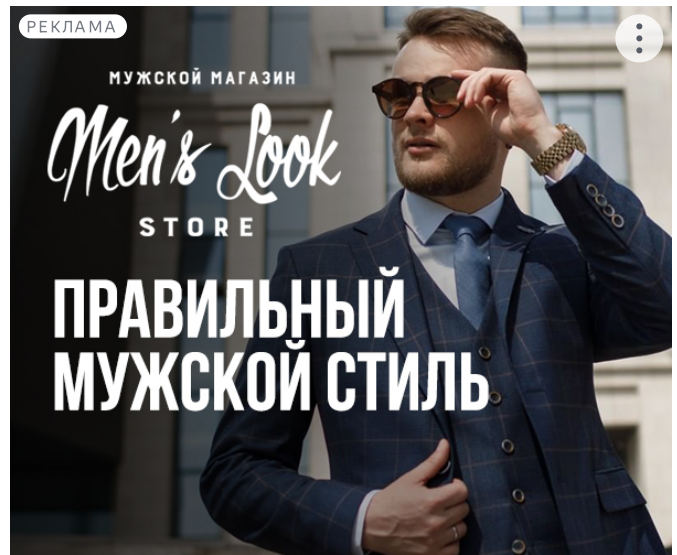
Для нахождения файлов нескольких типов надо писать так:

Так же можно в шаблоне указать путь:

Для того чтобы получить полный путь к документу используйте метод `realpath()`:

РЕКЛАМА

⋮



3. Информация об используемой памяти

Если вы будете отслеживать количество памяти, которое съедается на работу ваших скриптов то, наверное, чаще будете их оптимизировать.

В PHP существует мощный инструмент отслеживания используемой памяти. В разных частях скрипта нагрузки могут быть разные. Для того чтобы получить значение используемой памяти в данный момент, нам следует использовать метод `memory_get_usage()`. Для фиксации максимального количества используемой памяти используем `memory_get_peak_usage()`

4. Информация о процессоре

Для этого необходимо использовать метод `getrusage()`. Но учтите, что на Windows эта функция работать не будет.

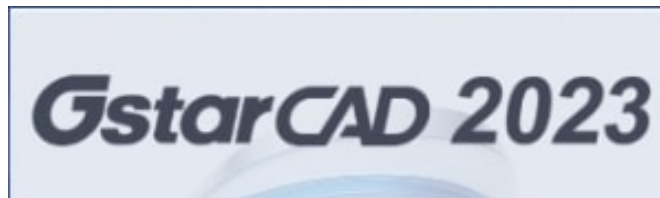
Картина, изложенная выше, будет понятно тем, у кого есть опыт в системном администрировании. Для всех остальных предлагаем расшифровку:

РЕКЛАМА



РЕКЛАМА





- `ru_oublock`: количество операций блочной записи
- `ru_inblock`: количество операций блочного чтения
- `ru_msgsnd`: количество отправленных сообщений
- `ru_msgrcv`: количество принятых сообщений
- `ru_maxrss`: максимальный размер невыгружаемого набора
- `ru_ixrss`: общий объем разделяемой памяти
- `ru_idrss`: общий объем неразделяемых данных
- `ru_minflt`: количество используемых страниц памяти
- `ru_majflt`: количество ошибок отсутствия страниц
- `ru_nsignals`: количество принятых сигналов
- `ru_nvcsw`: количество переключений контекста процессом
- `ru_nivcsw`: количество принудительных переключений контекста
- `ru_nswar`: количество обращений к диску при подкачке страниц
- `ru_utime.tv_usec`: время работы в пользовательском режиме (микросекунды)
- `ru_utime.tv_sec`: время работы в пользовательском режиме (секунды)
- `ru_stime.tv_usec`: время работы в привилегированном режиме (микросекунды)
- `ru_stime.tv_sec`: время работы в привилегированном режиме (секунды) РЕКЛАМА

⋮

Для того чтобы узнать какие ресурсы вашего процессора используются скриптом, вам необходимо значение `'user time'` (время работы в пользовательском режиме) и `'system time'` (время работы в привилегированном режиме). Вы можете получить результат как в секундах, так и в микросекундах. Для того чтобы превратить общее количество секунд в десятичное число, вам необходимо разделить значение микросекунд на 1 миллион и добавить к значению секунд.

Запутанно как-то. Вот пример:

Хотя выполнение скрипта заняло около 3-х секунд, процессор не был сильно нагружен. Дело в том, что при вызове (sleep) скрипт практически не потребляет ресурсов процессора. Вообще существует множество задач, которые занимают значительное время, но при этом не используют процессор. К примеру, ожидание операций связанных с диском. Так что вы не всегда используете процессорное время в своих скриптах.

РЕКЛАМА • 16+

Обучение JavaScript-разработке. Курс Яндекс Практикума

→

РЕКЛАМА

Работа скрипта заняла 1.4 секунды процессорного времени. В данном случае, время системных вызовов вообще низкое.

Время работы в привилегированном режиме (System Time) – это время, которое процессор затрачивает на выполнение системных запросов к ядру от имени программы. Пример:

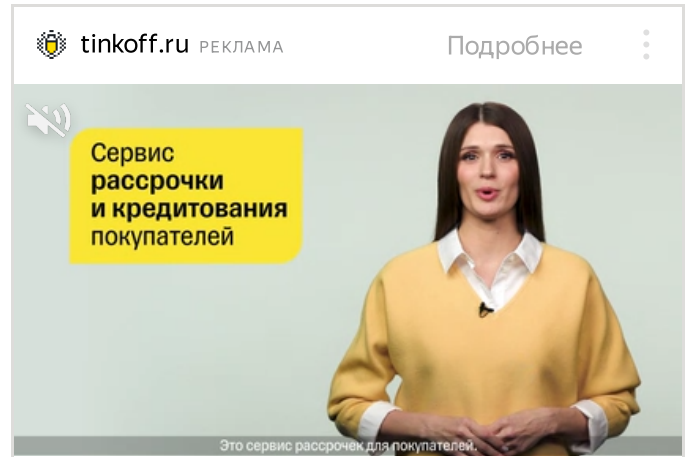
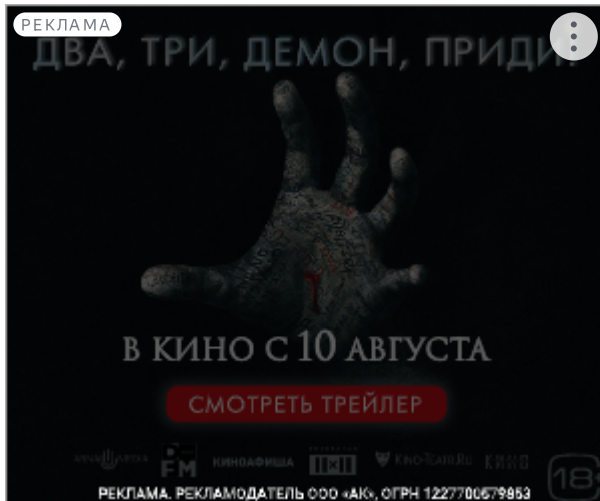
Теперь системного времени затратилось намного больше, чем в прошлом примере. Всё благодаря методу `microtime()`, который использует ресурсы системы.

Однако следует отметить, что выведенное время может быть не точным, т.к. в данный момент времени ресурсы процессора используются и другими программами, что в результате может дать небольшую погрешность.

5. Магические константы

В PHP существует множество магических констант, таких как номер текущей строки (`__LINE__`), путь к файлу (`__FILE__`), путь к каталогу (`__DIR__`), имя функции (`__FUNCTION__`), имя класса (`__CLASS__`), имя метода (`__METHOD__`) и пространства имён

(__NAMESPACE__).



Все мы их рассматривать не будем. Посмотрим только лишь парочку:

Используйте `__LINE__` при отладке скриптов:

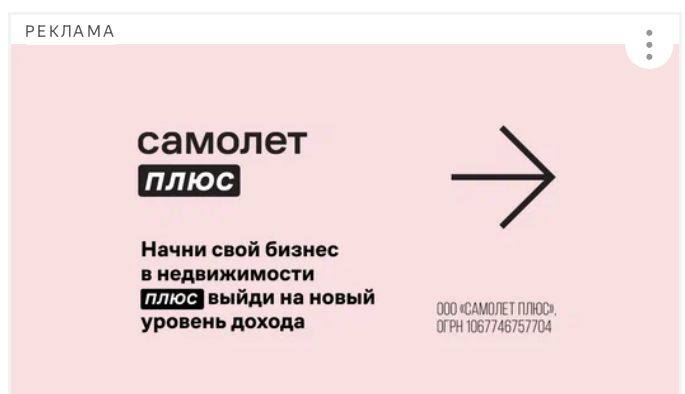
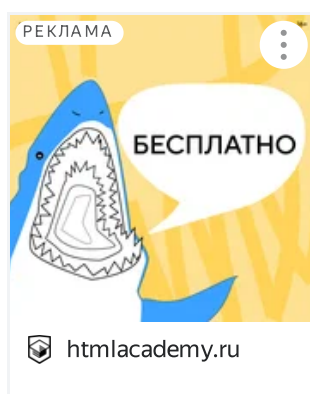
6. Генерирование уникальных ID

Бывают такие моменты, когда вам надо сгенерировать уникальную строку. Множество раз я видел, что для решения этой задачи используют функцию `md5()`:

Читайте также: Бюджетные беспроводные наушники вкладыши рейтинг

Но на самом деле для этих целей в PHP есть специальная функция `uniqid()`

Невооружённым взглядом можно заметить, что первые символы мягко говоря схожи... Так происходит из-за того, что данный метод использует время сервера для генерации символов. Это даже полезно, т.к. все сгенерированные значения получаются в алфавитном порядке, что даёт возможность быстро их сортировать.



**Онлайн-самоучитель
HTML и CSS.
Бесплатные
тренажеры**

Узнать больше

**Откройте агентство
в своем городе. Франшиза
Самолет Плюс**

→

Для того чтобы уменьшить шансы получения дубликата, мы можем добавить префикс или использовать второй параметр (увеличит количество символов):

Этот метод генерирует строки размером меньше, чем md5, тем самым вы сможете сэкономить место.

7. Сериализация

Вам когда-нибудь приходилось хранить комплексные данные в базе или в файле? Для того чтобы сконвертировать объект в строку в PHP предусмотрена специальная функция.

Вообще говоря, этих методов 2: `serialize()` и `unserialize()`

Вот так вот работают эти функции. Однако из-за бурного роста популярности JSON, в PHP 5.2 были добавлены 2 метода `json_encode()` и `json_decode()`. Их работа схожа с `serialize()`:

Этот вариант более компактный и совместимый с другими языками, такими как JavaScript. Однако при работе с очень навороченными объектами может возникнуть потеря данных.

РЕКЛАМА

БЕСПЛАТНО

htmlacademy.ru

**Онлайн-самоучитель
HTML и CSS.
Бесплатные
тренажеры**

Узнать больше

kontur-extern.ru РЕКЛАМА Go to site

РЕКЛАМА
КонтурЭкстерн

**Все
контролирующие
органы**

Росстат

СФР

→

ФНС

РПН

ЦБ РФ

8. Сжатие строк

Когда мы говорим о сжатии, то на ум сразу же приходят архивные файлы в формате ZIP. PHP предоставляет возможность сжатия длинных строк без всяких файлов.

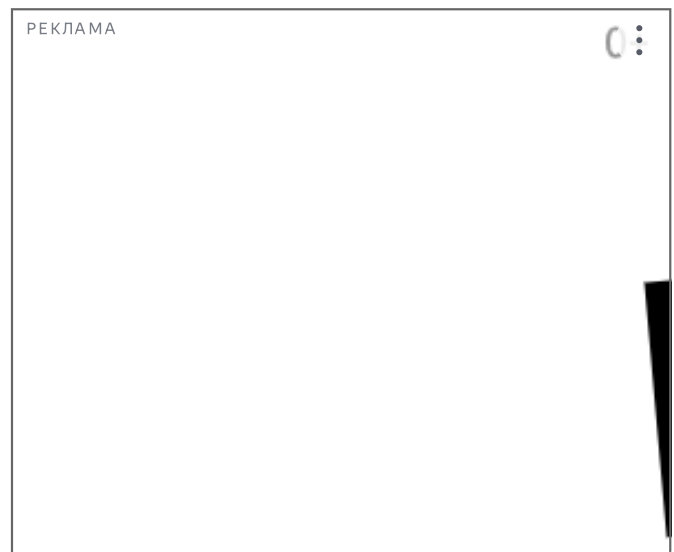
В следующем примере продемонстрируем работу функций `gzcompress()` и `gzuncompress()`:

В наших силах уменьшить объём текста на 50%. В этих же целях можно использовать методы `gzencode()` и `gzdecode()`, которые используют другой алгоритм сжатия.

9. Выполнить перед завершением

В PHP существует функция `register_shutdown_function()`, которая позволит вам выполнить какой-то код перед завершением работы скрипта.

Допустим, вы хотите узнать какую-то информацию... Время работы скрипта:



На первый взгляд это может показаться тривиальной задачей. Для этих целей, вы можете поместить код в конце файла. Однако если перед этим где-то сработает функция `exit()`, этот код никогда не сработает. Так же, он не сработает если на странице будет ошибка или пользователь прервёт загрузку страницы (нажав на соответствующую кнопку в своём браузере);

При использовании метода `register_shutdown_function()` код выполнится в любом случае:

Вывод

PHP это целая планета, которая не перестаёт нас удивлять своим содержимым. А что думаете вы о данных функциях?

Данный урок подготовлен для вас командой сайта ruseller.com

Источник урока: www.net.tutsplus.com/tutorials/php/9-useful-php-functions-and-features-you-need-to-know/

Перевел: Станислав Протасевич


Урок создан: 4 Марта 2011

Просмотров: 97126

Правила перепечатки

5 последних уроков рубрики «PHP»

РЕКЛАМА

The poster for the HOLY JS/... conference features a dark background with vibrant pink and purple abstract shapes. The text 'HOLY JS/...' is prominently displayed in a white, stylized font. Below it, the dates '2-3 ноября' and '11-12 ноября' are listed, along with 'ONLINE' and 'САНКТ-ПЕТЕРБУРГ + ONLINE' options. The website 'holyjs.ru' is at the bottom right.

Конференция для JS-разработчиков. 4 дня докладов

[→](#)

mvideo.ru РЕКЛАМА

Подробнее

The advertisement for mvideo.ru has a clean white background. It features the mvideo.ru logo and the text 'РЕКЛАМА. РЕКЛАМОДАТЕЛЬ ООО «МВМ» ИНН 7707548740'. The main headline 'ДЛЯ ВАШЕГО КОМФОРТА' is in large, bold, red letters. At the bottom, there is a small line of text about a promotion: 'В ПЕРИОД С 15.07.2023 ПО 14.08.2023 ПРОВОДИТСЯ АКЦИЯ «МЕРСИ» СКИДКИ ДО 60% НА ТОВАРЫ МЕЛКОЙ БЫТОВОЙ ТЕХНИКИ И ПРИЗЫ...



РЕКЛАМА



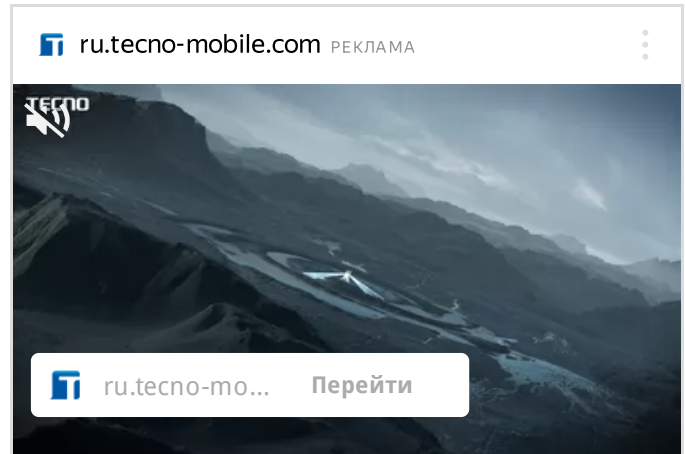
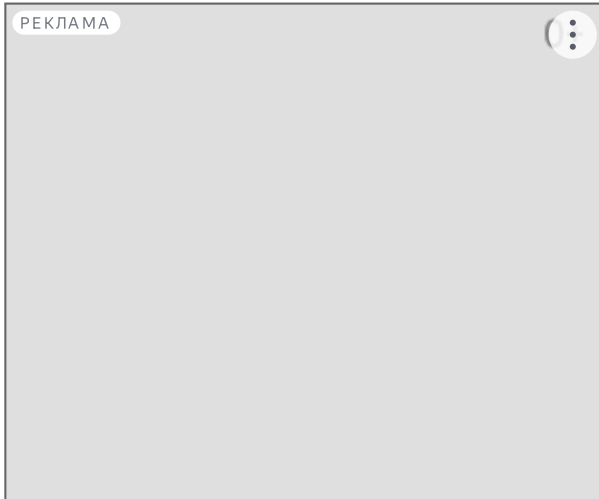
Фильтрация данных с помощью zend-filter

Когда речь идёт о безопасности веб-сайта, то фраза «фильтруйте всё, экранируйте всё» всегда будет актуальна. Сегодня поговорим о фильтрации данных.



Контекстное экранирование с помощью zend-escaper

Обеспечение безопасности веб-сайта — это не только защита от SQL инъекций, но и протекция от межсайтового скриптинга (XSS), межсайтовой подделки запросов (CSRF) и от других видов атак. В частности, вам нужно очень осторожно подходить к формированию HTML, CSS и JavaScript кода.



Подключение Zend модулей к Expressive

Expressive 2 поддерживает возможность подключения других ZF компонент по специальной схеме. Не всем нравится данное решение. В этой статье мы расскажем как улучшили процесс подключение нескольких модулей.

РЕКЛАМА



Совет: отправка информации в Google Analytics через API

Предположим, что вам необходимо отправить какую-то информацию в Google Analytics из серверного скрипта. Как это сделать. Ответ в этой заметке.

Подборка PHP песочниц

Подборка из нескольких видов PHP песочниц. На некоторых вы в режиме online сможете потестить свой код, но есть так же решения, которые можно внедрить на свой сайт.

[Источник](#)

9 полезных функций PHP и особенностей, которые нужно знать

РЕКЛАМА

⋮

Дата публикации: 2010-07-12

От Автора:

Даже после многолетнего использования PHP мы натыкаемся на функции и особенности, о которых не знали. Некоторые из них могут быть очень полезными, но мало используемыми. Не все из нас читали руководство и справочник функций от корки до корки!

1. Функции с произвольным количеством параметров

Вы могли уже узнать, что PHP позволяет определять функции с необязательными параметрами. Однако существует способ, разрешающий совершенно произвольное количество параметров у функции.

РЕКЛАМА

⋮

Для начала, вот пример исключительно с необязательными параметрами:

Бесплатный курс по PHP программированию

Освойте курс и узнайте, как создать веб-приложение на PHP с полного нуля

Теперь давайте посмотрим, как можно создать функцию, которая принимает любое количество параметров. На этот раз мы собираемся использовать `func_get_args()`:

2. Использование `glob()` для поиска файлов

У многих функций PHP длинные и содержательные имена. Однако будет трудно сказать, что делает функция с именем `glob()`, если вы уже откуда-то не знакомы с этим термином.

Думайте о ней как о более усовершенствованной версии функции `scandir()`. Она позволяет искать файлы, используя шаблоны.

Вы можете задавать несколько видов файлов, как здесь:

Обратите внимание, что файлы в действительности будут возвращаться с путями, зависящими от вашего запроса:

Если вам нужен полный путь к каждому файлу, просто вызовите функцию `realpath()` для массива возвращенных значений:

3. Сведения об использовании памяти

Обратив внимание на использование памяти своих скриптов, вы, возможно, сможете больше оптимизировать код.

PHP есть «сборщик мусора» (программа очистки памяти) и довольно сложный диспетчер памяти. Количество памяти, используемой вашим скриптом, может увеличиваться и уменьшаться в процессе его выполнения. Получить сведения о текущем использовании памяти можно, используя функцию `memory_get_usage()`, а чтобы узнать о наибольшем объеме памяти в любой точке, можно воспользоваться функцией `memory_get_peak_usage()`.

РЕКЛАМА

⋮

4. Сведения об использовании CPU

Для этого мы будем пользоваться функцией `getrusage()`. Помните о том, что она недоступна на платформе Windows.

Может выглядеть таинственно, пока у вас не будет навыков в администрировании системы. Вот объяснение каждого значения (учить это наизусть не нужно):

`ru_oublock`: количество операций блочной записи

`ru_inblock`: количество операций блочного чтения

`ru_msgsnd`: количество отправленных сообщений

`ru_msgrcv`: количество принятых сообщений

`ru_maxrss`: максимальный размер невыгружаемого набора

`ru_ixrss`: общий объем разделяемой памяти

`ru_idrss`: общий объем неразделяемых данных

`ru_minflt`: количество используемых страниц памяти

`ru_majflt`: количество ошибок отсутствия страниц

РЕКЛАМА



`ru_nsignals`: количество принятых сигналов

`gu_nvcsw`: количество переключений контекста процессом

`gu_nivcsw`: количество принудительных переключений контекста

`gu_nswar`: количество обращений к диску при подкачке страниц

`gu_utime.tv_usec`: время работы в пользовательском режиме (микросекунды)

`gu_utime.tv_sec`: время работы в пользовательском режиме (секунды)

`gu_stime.tv_usec`: время работы в привилегированном режиме (микросекунды)

`gu_stime.tv_sec`: время работы в привилегированном режиме (секунды)

Чтобы узнать, какие ресурсы CPU потребляет скрипт, нам нужно посмотреть на значения `'user time'` (время работы в пользовательском режиме) и `'system time'` (время работы в привилегированном режиме). По умолчанию величины секунд и миллисекунд представляются отдельно. Таким образом, вы можете разделить значение микросекунд на 1 миллион и добавить к значению секунд, чтобы вычислить общее количество секунд как десятичное число.

РЕКЛАМА

⋮

Хотя выполнение скрипта заняло примерно 3 секунды, использование CPU было очень-очень низким. Это происходит потому, что во время ожидания (`sleep`) скрипт фактически не потребляет ресурсов CPU. Существует множество других задач, которые могут занять реальное время, но при этом не использовать время CPU, например ожидание дисковой операции. Так что, как вы видите, использование CPU и действительная длительность времени исполнения – не всегда одно и то же.

Вот другой пример:

Бесплатный курс по PHP программированию

Освойте курс и узнайте, как создать веб-приложение на PHP с полного нуля

Этот скрипт использовал примерно 1,4 секунды времени CPU, и почти все время в пользовательском режиме, так как системных вызовов не было.

Время работы в привилегированном режиме (System Time) – это количество времени, которое CPU тратит на выполнение системных запросов к ядру от имени программы. Вот пример этого:

РЕКЛАМА

⋮

Теперь у нас используется довольно много времени привилегированного режима. Это происходит по тому, что скрипт много раз вызывает функцию `microtime()`, которая, чтобы получить данные о времени, выполняет запросы к ядру операционной системы.

Еще вы можете заметить, что цифры соответствуют 3 секундам неточно. Это потому что на сервере также, возможно, выполнялись и другие процессы, так что скрипт не использовал 100% CPU в течение всех 3 секунд.

5. Предопределенные, или «волшебные» константы

PHP предусматривает полезные «волшебные» константы для выборки текущего номера строки (`__LINE__`), пути файла (`__FILE__`), пути каталога (`__DIR__`), имени функции (`__FUNCTION__`), имени класса (`__CLASS__`), имени метода (`__METHOD__`) и пространства имен (`__NAMESPACE__`).

В этой статье мы не собираемся охватывать каждую из них, но я покажу вам некоторые случаи их использования.

Когда в дело включаются другие скрипты, хорошая идея — использование константы `__FILE__` (или `__DIR__` в версии PHP 5.3):

Источник

РЕКЛАМА

⋮

РЕКЛАМА

