

# Методические указания по теме “Встроенные функции PHP”

## Функции для работы со строками

[Функции для работы с символами](#)

[Функции для поиска в строке](#)

[Замена в тексте](#)

[Преобразование регистра](#)

[Работа с HTML-кодом](#)

[Экранирование](#)

[Локальные настройки \(локаль\)](#)

[Форматный вывод](#)

[Хранение данных](#)

[Работа с путями к файлам и каталогам](#)

[Объединение и разбиение строк](#)

## Функции даты и времени

[Формирование даты и времени](#)

[Форматирование даты и времени](#)

## Математические функции

[Поиск максимума и минимума](#)

[Генерация случайных чисел](#)

[Преобразование значений между разными системами счисления](#)

[Округление чисел](#)

[Логарифмические и степенные функции](#)

## Работа с файлами

[Создание файлов](#)

[Манипуляция файлами](#)

[Чтение и запись файлов](#)

[Права доступа к файлу](#)

[Работа с каталогами](#)



## Функции для работы со строками

### Функции для работы с символами

Функция	Описание
<code>strlen(\$str)</code>	Возвращает количество символов в строке <code>\$str</code>
<code>chr(\$ascii)</code>	Принимает в качестве аргумента ASCII-код <code>\$ascii</code> символа и возвращает соответствующий этому коду фактический символ
<code>ord(\$str)</code>	Возвращает ASCII-код символа <code>\$str</code> , переданного ей в качестве аргумента
<code>count_chars(\$str [, \$mode])</code>	Подсчитывает количество вхождений каждого из символов с ASCII-кодами в диапазоне 0...255 в строку <code>\$str</code> . Необязательный параметр <code>\$mode</code> позволяет задать формат результата
<code>str_shuffle(\$str)</code>	Возвращает копию строки <code>\$str</code> , символы которой перемешаны в случайном порядке
<code>strrev(\$str)</code>	Возвращает копию строки <code>\$str</code> , символы которой перевернуты, т. е. начальные символы помещены в конец, а конечные символы в начало
<code>ctype_alnum(\$str)</code>	Возвращает TRUE, если строка <code>\$str</code> содержит только буквы английского алфавита и цифры, в противном случае возвращается FALSE
<code>ctype_alpha(\$str)</code>	Возвращает TRUE, если строка <code>\$str</code> содержит только буквы английского алфавита, в противном случае возвращается FALSE
<code>ctype_xdigit(\$str)</code>	Возвращает TRUE, если строка <code>\$str</code> содержит только цифры и буквы A, B, C, D, E, F в верхнем или нижнем регистрах (т. е. символы, из которых состоят шестнадцатеричные числа). В противном случае возвращается FALSE
<code>ctype_cntrl(\$str)</code>	Возвращает TRUE, если строка <code>\$str</code> содержит только управляющие символы (перевод строки, символ табуляции и т. п.), в противном случае возвращается FALSE
<code>ctype_digit(\$str)</code>	Возвращает TRUE, если строка <code>\$str</code> содержит только цифры, в противном случае возвращается FALSE
<code>ctype_print(\$str)</code>	Возвращает TRUE, если строка <code>\$str</code> содержит только видимые символы, в противном случае возвращается FALSE
<code>ctype_graph(\$str)</code>	Возвращает TRUE, если строка <code>\$str</code> содержит только видимые графические символы, в противном случае возвращается FALSE. Пробел, символ табуляции традиционно относятся к видимым символам в



	таблице ASCII-кодов, однако не входят в группу графических символов — для них функция <code>ctype_graph()</code> вернет <code>FALSE</code>
<code>ctype_punct(\$str)</code>	Возвращает <code>TRUE</code> , если строка <code>\$str</code> содержит только символы пунктуации (видимые символы за исключением символов английского алфавита, цифр и пробельных символов), в противном случае возвращается <code>FALSE</code>
<code>ctype_lower(\$str)</code>	Возвращает <code>TRUE</code> , если строка <code>\$str</code> содержит только буквы английского алфавита в нижнем регистре, в противном случае возвращается <code>FALSE</code>
<code>ctype_upper(\$str)</code>	Возвращает <code>TRUE</code> , если строка <code>\$str</code> содержит только буквы английского алфавита в верхнем регистре, в противном случае возвращается <code>FALSE</code>

## Функции для поиска в строке

Задача поиска в строках является очень распространенной при создании Web-приложений. Существуют два подхода к решению проблемы: при помощи строковых функций и регулярных выражений. Регулярные выражения позволяют решать более сложные задачи, и в то же время они менее производительны, т. к. поиск с использованием регулярных выражений требует значительных расчетов. В таблице приводится список строковых функций, осуществляющих поиск в строке.

Функция	Описание
<code>substr(\$str, \$start [, \$length])</code>	Возвращает для строки <code>\$str</code> подстроку, которая начинается с символа <code>\$start</code> (отсчет с нуля) и имеет <code>\$length</code> символов. Если количество символов <code>\$length</code> не указано, предполагается, что подстрока должна включать все символы до конца строки <code>\$str</code>
<code>substr_count(\$str, \$search [, \$offset [, \$length]])</code>	Возвращает количество вхождений подстроки <code>\$search</code> в строку <code>\$str</code> . Поиск осуществляется с позиции <code>\$offset</code> в подстроке, длиной <code>\$length</code> (если эти два параметра указаны)
<code>strpos(\$str, \$search [, \$offset])</code>	Возвращает позицию первого вхождения подстроки <code>\$search</code> в строку <code>\$str</code> . Поиск может начинаться с позиции <code>\$offset</code> , если указывается третий параметр. Возвращает <code>FALSE</code> , если подстрока <code>\$search</code> не найдена
<code>stripos(\$str, \$search [, \$offset])</code>	Возвращает позицию первого вхождения подстроки <code>\$search</code> с конца строки <code>\$str</code> . Необязательный параметр <code>\$offset</code> позволяет задать позицию с конца строки, начиная с которой следует осуществлять поиск. Возвращает <code>FALSE</code> , если подстрока <code>\$search</code> не



	найдена
strstr(\$str, \$search [, \$before])	Ищет первое вхождение подстроки \$search в строку \$str. По умолчанию функция возвращает подстроку, начиная с первого вхождения \$search до конца строки \$str, однако если необязательный параметр \$before принимает параметр TRUE, возвращается подстрока, предшествующая первому вхождению подстроки \$search. Возвращает FALSE, если подстрока \$search не найдена
stristr(\$str, \$search [, \$before])	Функция stristr() аналогична функции strstr(), за исключением того факта, что в процессе поиска не учитывается регистр
strchr()	Синоним функции strstr()
strrchr(\$str, \$search)	Ищет последнее вхождение подстроки \$search в строку \$str и возвращает подстроку, начиная с \$search и до конца строки \$str. Возвращает FALSE, если подстрока \$search не найдена
strpbrk(\$str, \$char_list)	Ищет в строке \$str символы из строки \$char_list и возвращает подстроку, начиная с первого найденного символа и до конца строки. Если ни один символ не найден, возвращается FALSE

## Замена в тексте

Функции замены осуществляют преобразование строк, связанные с заменой одних подстрок другими, а также удалением подстрок.

Функция	Описание
str_replace(\$search, \$replace, \$str [, &\$count])	Возвращает копию строки \$str, в которой подстрока \$search заменена на \$replace. В необязательном параметре \$count может быть возвращено количество осуществленных замен. Вместо отдельных строк, параметры \$search и \$replace могут содержать массивы строк
str_ireplace(\$search, \$replace, \$str [, &\$count])	Функция str_ireplace() аналогична функции str_replace() за исключением того, что поиск осуществляется без учета регистра
substr_replace(\$str, \$replacement, \$start [, \$length])	Возвращает копию строки \$str, в которой подстрока, начинающаяся с символа \$start и длиной \$length, заменена подстрокой replacement

strtr(\$str, \$from, \$to)	Возвращает копию строки \$str, в которой каждый символ из строки \$from заменен соответствующим символом из строки \$to
strtr(\$str, \$replace_pairs)	Возвращает копию строки \$str, в которой замены осуществлены в соответствии с массивом \$replace_pairs (ключи заменяются значениями массива)
trim(\$str [, \$charlist])	Удаляет пробельные символы из начала и конца строки \$str
rtrim(\$str [, \$charlist])	Удаляет пробельные символы из конца строки \$str. Необязательный параметр \$charlist позволяет уточнить список пробельных символов
chop()	Синоним для функции rtrim()
ltrim(\$str [, \$charlist])	Удаляет пробельные символы из начала строки \$str. Необязательный параметр \$charlist позволяет уточнить список пробельных символов

## Преобразование регистра

Функция	Описание
strtolower(\$str)	Преобразует строку \$str в нижний регистр
strtoupper(\$str)	Преобразует строку \$str в верхний регистр
ucfirst(\$str)	Преобразует первый символ строки \$str в верхний регистр
ucwords(\$str)	Преобразует каждое слово строки в верхний регистр

## Работа с HTML-кодом

Так как PHP проектировался как язык, специально предназначенный для Web, среди строковых функций имеется несколько, специально предназначенных для Web-разработки.

Функция	Описание
nl2br(\$str)	Добавляет перед переводами строки HTML-тег перевода строки  
htmlspecialchars(\$str [, \$quote_style [, \$charset [,	Преобразует HTML-теги в строке \$str в

<code>\$double_encode]]])</code>	представление, делающее их видимыми в браузере. Необязательный параметр <code>\$quote_style</code> определяет режим замены кавычек и может принимать три значения: <code>ENT_NOQUOTES</code> — двойные и одиночные кавычки остаются без изменений, <code>ENT_COMPAT</code> — двойные кавычки преобразуются, одиночные остаются без изменений, <code>ENT_QUOTES</code> — одиночные и двойные кавычки преобразуются. Параметр <code>\$charset</code> задает кодировку текста. Если четвертый необязательный параметр <code>\$double_encode</code> принимает значение <code>FALSE</code> , один раз уже преобразованный HTML-тег не подвергает вторичному преобразованию функции. По умолчанию, этот параметр принимает значение <code>TRUE</code>
<code>htmlspecialchars_decode(\$str [, \$quote_style])</code>	Выполняет обратную функции <code>htmlspecialchars()</code> задачу — преобразует представление HTML-тегов в исходные HTML-теги
<code>htmlentities(\$str [, \$quote_style [, \$charset [, \$double_encode]]])</code>	Функция аналогична <code>htmlspecialchars()</code> , однако преобразует не минимальное количество символов для представления HTML-кода, а все возможные символы, для которых предусмотрено представление
<code>html_entity_decode(\$str [, \$quote_style [, \$charset]])</code>	Выполняет обратную функции <code>htmlentities()</code> задачу — преобразует представление HTML-тегов в исходные HTML-теги
<code>get_html_translation_table ( [\$table [, \$quote_style]])</code>	Возвращает таблицу преобразований, используемую функциями <code>htmlspecialchars()</code> и <code>htmlentities()</code> . Необязательный параметр <code>\$table</code> может принимать либо константу <code>HTML_ENTITIES</code> , либо <code>HTML_SPECIALCHARS</code> для <code>htmlspecialchars()</code> и <code>htmlentities()</code> , соответственно. Параметр <code>\$quote_style</code> определяет режим замены кавычек
<code>strip_tags(\$str [, \$allowable_tags])</code>	Удаляет из строки <code>\$str</code> HTML-теги, кроме тех, которые указываются в параметре <code>\$allowable_tags</code>

## Экранирование

Группа функций экранирования предназначена для экранирования специальных символов в строках.

Функция	Описание
addslashes(\$str)	Возвращает копию строки \$str, в которой перед каждым специальным символом добавлен обратный слеш (\). Экранируются одиночная кавычка ('), двойная кавычка ("), обратный слеш (\) и NUL (символ \0)
stripslashes(\$str)	Функция stripslashes() является обратной для addslashes(), предназначена для удаления обратных слешей
addcslashes(\$str, \$charlist)	Возвращает копию строки \$str, в которой перед каждым символом из перечисленных в \$charlist добавлен обратный слеш (\)
stripccslashes(\$str)	Функция stripccslashes() удаляет символ обратного слеша и является противоположной функции addcslashes()
quoted_printable_decode(\$str)	Декодирует строку \$str, закодированную методом quoted printable
quotemeta(\$str)	Возвращает копию строки \$str, в которой перед каждым символом . \ + * ? [ ^ ] ( \$ ) помещается обратный слеш (\). Функция удобна для экранирования данных в регулярных выражениях

## Локальные настройки (локаль)

В разных странах в силу сложившихся традиций имеются различия в представлении чисел, денежных сумм, времени и даты. Правила их представления для каждой страны называют локальными настройками или сокращенно локалью. Каждая операционная система, будь то Windows или одна из UNIX-подобных ОС, поддерживает локальные настройки, которые позволяют изменять представление данных. Такой возможностью обладает и PHP. Ниже представлены функции, ответственные за работу с локалью.

Функция	Описание
Setlocale(\$category, \$locale [, ...])	Устанавливает локальные настройки или возвращает текущую локальную настройку. Функция возвращает значение только что установленной локали или FALSE, если операционная система не поддерживает работу с локальными настройками
localeconv()	Возвращает ассоциативный массив с информацией о числовых и денежных форматах текущей локали. Ключи массива и их значения описываются в таблице ниже
nl_langinfo(\$item)	Предоставляет доступ к отдельным элементам локали. Если аргумент \$item имеет недопустимое значение, возвращает FALSE.

Локальные настройки управляются шестью переменными окружения, список которых представлен ниже в таблице.

Переменная окружения	Описание
LC_COLLATE	Определяет правила сравнения и, следовательно, сортировки строк для местного алфавита
LC_CTYPE	Определяет правила преобразования одиночных символов для местного алфавита. Позволяет правильно распознавать вид символа: цифра, буква, знак, верхний и нижний регистр
LC_MONETARY	Определяет правила национального представления денежных величин
LC_NUMERIC	Определяет правила национального представления чисел с плавающей точкой
LC_TIME	Определяет правила национального представления даты и времени. Задаёт именование дней недели, месяцев, формат даты
LC_MESSAGES	Определяет формат информационных, диагностических и интерактивных сообщений операционной системы
LC_ALL	Особая переменная окружения, позволяющая устанавливать значения для всех вышеперечисленных переменных

Функция `localeconv()` возвращает ассоциативный массив с информацией о числовых и денежных форматах. Структура результирующего массива описывается таблице ниже.

Ключ	Описание
<code>decimal_point</code>	Символ десятичной точки
<code>thousands_sep</code>	Разделитель групп (тысячи)
<code>grouping</code>	Массив, содержащий количество цифр в группах для числовых данных
<code>int_curr_symbol</code>	Международное обозначение валюты, например, RUR
<code>currency_symbol</code>	Национальное обозначение валюты, например, руб
<code>mon_decimal_point</code>	Символ десятичной точки в денежном формате
<code>mon_thousands_sep</code>	Разделитель групп в денежном формате
<code>mon_grouping</code>	Массив, содержащий количества цифр в группах для денежных данных
<code>positive_sign</code>	Знак для положительных чисел





negative_sign	Знак для отрицательных чисел
int_frac_digits	Количество разрядов после точки (международное)
frac_digits	Количество разрядов после точки (национальное)
p_cs_precedes	TRUE, если currency_symbol записывается перед положительным значением, иначе FALSE
p_sep_by_space	TRUE, если currency_symbol отделяется от положительного значения пробелом, иначе FALSE
n_cs_precedes	TRUE, если currency_symbol записывается перед отрицательным значением, иначе FALSE
n_sep_by_space	TRUE, если currency_symbol отделяется от отрицательного значения пробелом, иначе FALSE
p_sign_posn	Для положительных чисел: 0 — число и обозначение валюты заключаются в скобки; 1 — знак записывается перед числом и обозначением валюты; 2 — знак записывается после числа и обозначения валюты; 3 — знак записывается перед обозначением валюты; 4 — знак записывается после обозначения валюты
n_sign_posn	Для отрицательных чисел: 0 — число и обозначение валюты заключаются в скобки; 1 — знак записывается перед числом и обозначением валюты; 2 — знак записывается после числа и обозначения валюты; 3 — знак записывается перед обозначением валюты; 4 — знак записывается после обозначения валюты

## Форматный вывод

Функции данной группы осуществляют вывод информации в окно браузера и ее форматирование.

Функция	Описание
echo(\$arg1 [, ...])	Выводит аргумент \$arg1 и все последующие в окно браузера
print(\$arg)	Выводит аргумент \$arg в окно браузера
printf(\$format [, \$args [, ...]])	Выводит аргументы \$args в окно браузера, отформатированные в соответствии со строкой \$format
fprintf(\$handle, \$format [, \$args [, ...]])	Выводит аргументы \$args в открытый файл \$handle, отформатированные в соответствии со строкой \$format
sprintf(\$format [, \$args [, ...]])	Возвращает строку, состоящую из аргументов \$args, отформатированных в соответствии со строкой \$format

<code>vprintf(\$format, \$args)</code>	Выводит элементы массива <code>\$args</code> в окно браузера, отформатированные в соответствии со строкой <code>\$format</code>
<code>fprintf(\$handle, \$format, \$args)</code>	Выводит элементы <code>\$args</code> в открытый файл <code>\$handle</code> , отформатированные в соответствии со строкой <code>\$format</code>
<code>vsprintf(\$format, \$args)</code>	Возвращает строку, состоящую из элементов массива <code>\$args</code> , отформатированных в соответствии со строкой <code>\$format</code>
<code>number_format(\$number [, \$decimals [, \$dec_point, \$thousands_sep]])</code>	Возвращает отформатированное число <code>\$number</code> с <code>\$decimals</code> знаками после запятой. При этом в качестве десятичной точки будет использован символ <code>\$dec_point</code> , а тысячи будет разделять символ <code>\$thousands_sep</code>
<code>money_format(\$format, \$number)</code>	Возвращает отформатированную денежную сумму <code>\$number</code> в соответствии со строкой форматирования <code>\$format</code> и текущей локалью

## Хранение данных

Строки очень часто используются для хранения разнообразных данных. Поэтому PHP имеет целый арсенал функций для упаковки данных в строки и извлечению их.

Функция	Описание
<code>bin2hex(\$str)</code>	Осуществляет побайтовое преобразование символов строки <code>\$str</code> в шестнадцатеричный формат
<code>pack(\$format[, \$args [, ...]])</code>	Упаковывает параметры <code>\$args</code> в двоичную строку. Формат параметров и их количество определяется первым параметром <code>\$format</code>
<code>unpack(\$format, \$data)</code>	Осуществляет обратное функции <code>pack()</code> преобразование
<code>serialize(\$value)</code>	Осуществляет упаковку массива или объекта в строку <code>\$value</code>
<code>unserialize(\$str)</code>	Осуществляет распаковку массива или объекта из строки <code>\$str</code> , созданной функцией <code>serialize()</code>

## Работа с путями к файлам и каталогам

Функции данной группы позволяют облегчить работу с путями к файлам и каталогам.

Функция	Описание
<code>pathinfo(\$path [, \$options])</code>	Принимает путь к файлу <code>\$path</code> и возвращает ассоциативный

	<p>массив, в элементах которого сохраняются каталог, в котором расположен файл, имя файла, его расширение и имя файла без расширения. Если указан необязательный параметр <code>\$options</code>, функция возвращает строковое значение. Параметр <code>\$options</code> может принимать следующие константы:</p> <ul style="list-style-type: none"> <li>• <code>PATHINFO_DIRNAME</code> — путь к файлу (каталог);</li> <li>• <code>PATHINFO_BASENAME</code> — имя файла;</li> <li>• <code>PATHINFO_EXTENSION</code> — расширение файла;</li> <li>• <code>PATHINFO_FILENAME</code> — имя файла без расширения</li> </ul>
<code>realpath(\$path)</code>	Возвращает абсолютный путь (путь от начала диска) для файла <code>\$path</code>
<code>basename(\$path [, \$suffix])</code>	Извлекает из пути <code>\$path</code> имя файла. Необязательный параметр <code>\$suffix</code> позволяет исключить расширение из результирующей строки
<code>dirname(\$path)</code>	Извлекает из пути <code>\$path</code> каталог, в котором расположен файл

## Объединение и разбиение строк

Функции данной группы осуществляют объединение подстрок в единую строку, а также разбиение строки на отдельные подстроки.

Функция	Описание
<code>str_repeat(\$str, \$number)</code>	Создает новую строку, состоящую из <code>\$number</code> повторов строки <code>\$str</code>
<code>str_pad(\$str, \$length [, \$pad [, \$pad_type]])</code>	Увеличивает размер строки <code>\$str</code> до <code>\$length</code> символов. По умолчанию новые символы заполняются пробелами, однако символзаполнитель можно задать в необязательном параметре <code>\$pad</code> . Необязательный параметр <code>\$pad_type</code> определяет, с какой стороны добавляются новые символы, и может принимать следующие значения: <ul style="list-style-type: none"> <li>• <code>STR_PAD_RIGHT</code> — новые символы добавляются в конец строки (по умолчанию);</li> <li>• <code>STR_PAD_LEFT</code> — новые символы добавляются в начало строки;</li> <li>• <code>STR_PAD_BOTH</code> — новые символы добавляются как в конец, так и в начало;</li> </ul>
<code>chunk_split(\$str [, \$chunklen [, \$end]])</code>	Возвращает копию строки <code>\$str</code> , в которой через каждые <code>\$chunklen</code> (по умолчанию 76) символов вставляется подстрока <code>\$end</code> (по умолчанию <code>\r\n</code> )
<code>str_split(\$str [, \$split])</code>	Преобразует строку <code>\$str</code> в массив. По умолчанию в качестве элементов массива выступают символы строки.



	Необязательный параметр <code>\$split</code> позволяет задать произвольную длину подстрок в символах, образующих элементы массива
<code>strtok(\$str, \$token)</code>	Разбивает строку <code>\$str</code> на подстроки, используя в качестве разделителя подстрок последовательность <code>\$token</code>
<code>str_word_count(\$str [, \$format [, \$charlist]])</code>	Если указывается единственный параметр <code>\$str</code> , возвращается количество слов в строке. Если указывается необязательный параметр <code>\$format</code> , отдельные слова строки возвращаются в виде массива. Параметр <code>\$format</code> может принимать следующие значения: 1 — возвращается массив, содержащий все слова, входящие в строку <code>\$str</code> ; 2 — возвращается массив, индексами которого являются позиции в строке <code>\$str</code> , а значениями — соответствующие слова Необязательный параметр <code>\$charlist</code> позволяет задать дополнительные символы, из которых будет состоять слово
<code>explode(\$delimiter, \$str [, \$limit])</code>	Функция возвращает массив из строк, каждая из которых соответствует фрагменту исходной строки <code>\$str</code> , находящемуся между разделителями, определяемыми аргументом <code>\$delimiter</code> . Необязательный параметр <code>\$limit</code> определяет максимальное количество элементов в массиве
<code>implode(\$delimiter, \$arr)</code>	Объединяет элементы массива строк <code>\$arr</code> в единую строку, разделяя элементы подстрокой, заданной в параметре <code>\$delimiter</code>
<code>join()</code>	Синоним для функции <code>implode()</code>
<code>wordwrap(\$str [, \$width [, \$break [, \$cut]])</code>	Вставляет в строке <code>\$str</code> символы <code>\$break</code> (по умолчанию <code>\n</code> ) через каждые <code>\$width</code> символов (по умолчанию 75). При этом символ <code>\$break</code> не разрывает слово, однако если необязательный параметр <code>\$cut</code> принимает значение <code>TRUE</code> , допускается разрыв слова
<code>parse_str(\$str [, &amp;\$arr])</code>	Разбивает строку <code>\$str</code> , имеющую формат строки с GET-запросом, и преобразует отдельные GET-параметры в переменные или в элементы массива <code>\$arr</code> (если указан второй параметр)
<code>sscanf(\$str, \$format [, ...])</code>	Разбирает строку <code>\$str</code> в соответствии со строкой форматирования <code>\$format</code> . Если передано только два аргумента, функция возвращает массив, в противном случае результаты помещаются в последующие параметры



## Функции даты и времени

Исторически сложилось, что время и дата имеют собственную достаточно запутанную систему счисления, которая значительно отличается от десятичной, используемой в математических и компьютерных вычислениях. Чтобы совместить эти две системы счисления, в компьютерных вычислениях время хранят в UNIXSTAMP-формате, при необходимости форматируя дату для отображения в традиционном формате.

В UNIXSTAMP-формате время хранится в секундах, прошедших с 0:00 1 января 1970 года. Время в таком формате позволяет легко сравнивать даты друг с другом, а также прибавлять и вычитать интервалы времени. Функции, формирующие дату в этом формате, представлены.

### Формирование даты и времени

Функция	Описание
<code>time()</code>	Возвращает текущую дату и время в UNIXSTAMP-формате
<code>microtime([\$get_as_float])</code>	Возвращает текущую дату и время в UNIXSTAMP-формате с микросекундами. Если параметр <code>\$get_as_float</code> принимает значение <code>TRUE</code> , функция возвращает результат в виде числа с плавающей точкой
<code>mktime([\$hour [, \$minute [, \$second [, \$month [, \$day [, \$year [, \$is_dst]]]]]])</code>	Возвращает дату и время в UNIXSTAMP-формате с годом <code>\$year</code> , месяцем <code>\$month</code> , днем месяца <code>\$day</code> , часом <code>\$hour</code> , минутами <code>\$minute</code> , секундами <code>\$second</code> . Если параметр <code>\$is_dst</code> принимает значение <code>1</code> , считается, что установлено летнее время, если <code>0</code> — зимнее, если <code>-1</code> или параметр не указан вовсе, PHP пытается автоматически определить его
<code>gmmktime([\$hour [, \$minute [, \$second [, \$month [, \$day [, \$year [, \$is_dst]]]]]])</code>	Функция <code>gmmktime()</code> аналогична <code>mktime()</code> , однако возвращается не местное время, а время по Гринвичу
<code>gettimeofday([\$return_float])</code>	Возвращает текущую дату и время либо в виде ассоциативного массива, либо, если параметр <code>\$return_float</code> принимает значение <code>TRUE</code> , в виде числа с плавающей точкой
<code>strtotime(\$time [, \$timestamp])</code>	Разбирает строку с датой и временем и возвращает их в формате UNIXSTAMP

## Форматирование даты и времени

Функция	Описание
<code>date(\$format [, \$timestamp])</code>	Форматирует текущую дату в соответствии со строкой, переданной в качестве первого параметра <code>\$format</code> . Может отформатировать произвольную дату, если она передана в качестве второго необязательного параметра <code>\$timestamp</code>
<code>gmdate(\$format [, \$timestamp])</code>	Функция <code>gmdate()</code> аналогична функции <code>date()</code> , однако возвращает не текущее локальное время, а время по Гринвичу
<code>idate(\$format [, \$timestamp])</code>	Форматирует текущую дату в соответствии со строкой, переданной в качестве первого параметра <code>\$format</code> . Может отформатировать произвольную дату, если она передана в качестве второго необязательного параметра <code>\$timestamp</code> . В отличие от функции <code>date()</code> возвращает лишь целые значения
<code>checkdate(\$month, \$day, \$year)</code>	Проверяет, является ли дата с годом <code>\$year</code> , месяцем <code>\$month</code> и днем <code>\$day</code> корректной, и возвращает <code>TRUE</code> , если это так, и <code>FALSE</code> в противном случае
<code>getdate([\$timestamp])</code>	Возвращает текущую дату и время в виде ассоциативного массива. Через необязательный параметр <code>\$timestamp</code> можно передать произвольную временную метку
<code>strftime(\$format [, \$timestamp])</code>	Форматирует текущие дату и время согласно локальным настройкам. Параметр <code>\$format</code> позволяет задать строку форматирования. Необязательный параметр <code>\$timestamp</code> позволяет передать на обработку функции произвольную временную метку
<code>gmstrftime(\$format [, \$timestamp])</code>	Функция <code>gmstrftime()</code> аналогична функции <code>strftime()</code> , однако возвращает не текущее локальное время, а время по Гринвичу
<code>strtotime(\$date, \$format)</code>	Разбирает строку, сгенерированную функцией <code>strftime()</code> , и возвращает дату и время в виде ассоциативного массива
<code>date_parse(\$date)</code>	Разбирает строку, сгенерированную функцией <code>strftime()</code> , и возвращает дату и время в виде ассоциативного массива
<code>localtime([\$timestamp [, \$is_associative]])</code>	Возвращает текущую дату и время в виде индексного массива (или ассоциативного массива, если параметр <code>\$is_associative</code> принимает значение <code>TRUE</code> ). Массив может быть возвращен не только для текущего времени, но и для произвольной временной метки в формате <code>UNIXSTAMP</code> ,



которая может быть передана через параметр \$timestamp



## Математические функции

### Поиск максимума и минимума

Функция	Описание
<code>max(\$value1, [\$value2 [, \$value3...]])</code>	Возвращает максимальное значение среди параметров
<code>min(\$value1, [\$value2 [, \$value3...]])</code>	Возвращает минимальное значение среди параметров

### Генерация случайных чисел

Функция	Описание
<code>rand([\$min, \$max])</code>	Генерирует случайное число в интервале от \$min до \$max. Если параметры \$min и \$max отсутствуют, возвращается число, лежащее в пределах от 0 до RAND_MAX (32 768)
<code>mt_rand([\$min, \$max])</code>	Более быстрая версия функции rand(), обладающая более равномерным распределением случайных значений (что важно с точки зрения криптографии)
<code>lcg_value()</code>	Генерирует и возвращает случайное дробное число, лежащее в диапазоне от 0 до 1

### Преобразование значений между разными системами счисления

Функция	Описание
<code>base_convert(\$number, \$frombase, \$tobase)</code>	Переводит число \$number из системы счисления по основанию \$frombase в систему по основанию \$tobase. \$frombase и \$tobase принимают значения только от 2 до 36 включительно. В строке \$number цифры обозначают сами себя, символ a соответствует числу 11, b — числу 12 и т. д. Символ z обозначает число 36
<code>decbin(\$number)</code>	Преобразует десятичное число \$number в двоичное
<code>decoct(\$number)</code>	Преобразует десятичное число \$number в восьмеричное





dechex(\$number)	Преобразует десятичное число \$number в шестнадцатеричное
bindec(\$binary)	Преобразует двоичное число \$binary в десятичное
hexdec(\$hex)	Преобразует шестнадцатеричное число \$hex в десятичное
octdec(\$octal)	Преобразует восьмеричное число \$octal в десятичное
deg2rad(\$number)	Преобразует градусы \$number в радианы
rad2deg(\$number)	Преобразует радианы \$number в градусы

## Округление чисел

Функция	Описание
abs(\$number)	Возвращает модуль числа <i>\$number</i>
round(\$value [, \$precision])	Округляет число \$value, если указан необязательный параметр \$precision — округление осуществляется до \$precision знака после запятой
ceil(\$value)	Округляет число \$value до ближайшего целого числа, причем результат всегда больше \$value
floor(\$value)	Округляет число \$value до ближайшего целого числа, причем результат всегда меньше \$value
fmod(\$x, \$y)	Возвращает остаток деления двух чисел \$x и \$y

## Логарифмические и степенные функции

Функция	Описание
pow(\$base, \$exp)	Возвращает число \$base в степени \$exp
exp(\$number)	Возвращает экспоненту в степени \$number
expm1(\$number)	Возвращает экспоненту в степени \$number минус единица
sqrt(\$number)	Возвращает корень квадратный числа \$number
log(\$number [, \$base])	Возвращает логарифм числа \$number по основанию \$base, если параметр \$base не указан — возвращается натуральный логарифм

<code>log10(\$number)</code>	Возвращает десятичный логарифм числа <code>\$number</code>
<code>log1p(\$number)</code>	Возвращает натуральный логарифм от суммы <code>\$number</code> плюс единица



# Работа с файлами

## Создание файлов

PHP предоставляет разработчикам большое количество функций для работы с файлами. Первыми рассмотрим функции, позволяющие создать файлы.

Функция	Описание
<code>fopen(\$filename, \$mode [, \$use_include_path[, \$context]])</code>	Открывает файл с именем <code>\$filename</code> в режиме <code>\$mode</code> . Некоторые режимы позволяют создать файл, некоторые требуют, чтобы файл уже существовал. Необязательный параметр <code>\$use_include_path</code> позволяет задать путь для поиска файла. В случае успеха функция возвращает дескриптор открытого файла, в случае неудачи — <code>FALSE</code> . Параметр <code>\$context</code> позволяет задать HTTP-заголовки, отправляемые сервером при обращении к сетевому файлу.
<code>fclose(\$handle)</code>	Закрывает файл, открытый ранее при помощи функции <code>fopen()</code> . В качестве единственного параметра <code>\$handle</code> функция принимает дескриптор открытого файла. Возвращает <code>TRUE</code> при успешном закрытии файла и <code>FALSE</code> в противном случае
<code>touch(\$filename [, \$time [, \$atime]])</code>	Функция <code>touch()</code> предназначена для изменения времени последней модификации файла <code>\$time</code> и времени последнего доступа <code>\$atime</code> . Если файл не существует — он создается, поэтому функция часто используется для создания файлов
<code>tempnam(\$dir, \$prefix)</code>	Создает в каталоге <code>\$dir</code> файл с уникальным именем, при этом в имени файла используется префикс <code>\$prefix</code> . В случае успеха возвращает имя нового файла, в случае неудачи — <code>FALSE</code>
<code>tmpfile()</code>	Создает временный файл с уникальным именем и возвращает его дескриптор. Файл автоматически удаляется после его закрытия. Если создать файл не удалось, функция возвращает <code>FALSE</code>

## Манипуляция файлами

Помимо операции создания файлов очень часто возникают задачи, связанные с их перемещением, переименованием и удалением. Функции, ответственные за эти операции, представлены в таблице.

Функция	Описание
<code>copy(\$source, \$destination)</code>	Копирует файл с именем <code>\$source</code> в файл с именем <code>\$destination</code> . В случае успешного копирования функция возвращает <code>TRUE</code> , в противном случае возвращает <code>FALSE</code>
<code>unlink(\$filename)</code>	Удаляет файл с именем <code>\$filename</code> . В случае успешного удаления функция возвращает <code>TRUE</code> , в противном случае возвращает <code>FALSE</code>
<code>rename(\$oldname, \$newname)</code>	Переименовывает файл с именем <code>\$oldname</code> , назначая ему новое имя <code>\$newname</code> . В случае успешного переименования функция возвращает <code>TRUE</code> , в противном случае возвращает <code>FALSE</code>
<code>is_uploaded_file(\$filename)</code>	Возвращает <code>TRUE</code> , если файл был загружен на сервер, и <code>FALSE</code> в противном случае. В качестве аргумента функция принимает элемент массива <code>\$_FILES['filename']['tmp_name']</code> , содержащий загруженный файл во временном каталоге
<code>move_uploaded_file (\$filename, \$destination)</code>	Перемещает файл из временного каталога в каталог назначения. В качестве первого аргумента <code>\$filename</code> зачастую используется элемент <code>\$_FILES['filename']['tmp_name']</code> , второй аргумент может быть произвольным. Если необходимо сохранить исходное имя файла, можно воспользоваться элементом <code>\$_FILES['filename']['name']</code> . В отличие от функции <code>copy()</code> функция <code>move_uploaded_file()</code> оперирует лишь загруженными файлами

## Чтение и запись файлов

Данные функции раздела позволяют оперировать содержимым файла: читать, записывать и перезаписывать.

Функция	Описание
<code>fopen(\$filename, \$mode [, \$use_include_path [, \$context]])</code>	Открывает файл с именем <code>\$filename</code> в режиме <code>\$mode</code> . Некоторые режимы позволяют создать файл, некоторые требуют, чтобы файл уже существовал. Необязательный параметр <code>\$use_include_path</code> позволяет задать путь для поиска файла. В случае успеха функция возвращает дескриптор открытого файла, в случае неудачи — <code>FALSE</code> . Параметр <code>\$context</code> позволяет задать HTTP-заголовки, отправляемые сервером при обращении к сетевому файлу



<code>fclose(\$handle)</code>	Закрывает файл, открытый ранее при помощи функции <code>foren()</code> . В качестве единственного параметра <code>\$handle</code> функция принимает дескриптор открытого файла. Возвращает <code>TRUE</code> при успешном закрытии файла и <code>FALSE</code> в противном случае
<code>feof(\$handle)</code>	Возвращает <code>TRUE</code> , если файловый указатель <code>\$handler</code> указывает на конец файла, и <code>FALSE</code> в противном случае
<code>fgetc(\$handle)</code>	Считывает и возвращает в качестве результата из открытого файла <code>\$handle</code> один символ
<code>fgets(\$handle [, \$length])</code>	Считывает и возвращает в качестве результата из открытого файла <code>\$handle</code> строку. Чтение заканчивается по достижении строки длины <code>\$handle - 1</code> (по умолчанию <code>\$handle</code> принимает значение 1000), по достижении символа перевода строки или символа конца файла
<code>fgetss(\$handle [, \$length [, \$allowable_tags]])</code>	Аналогична функции <code>fgets()</code> , однако из прочитанной строки удаляются все HTML-теги. Необязательный параметр <code>\$allowable_tags</code> может содержать допустимые HTML-теги, которые не должны отбрасываться
<code>fread(\$handle, \$length)</code>	Считывает и возвращает в качестве результата из открытого файла <code>\$handle</code> строку длиной <code>\$length</code> символов. В отличие от функции <code>fgets()</code> переводы строк не прекращают чтение
<code>fscanf(\$handle, \$format [, ...])</code>	Аналогична функции форматного разбора строки <code>sscanf()</code> , однако чтение и разбор строки по форматному правилу <code>\$format</code> осуществляются не из строки, а из открытого файла <code>\$handler</code> . Если в функцию переданы только два первых параметра, обработанные значения будут возвращены в виде массива. В противном случае, если были переданы необязательные аргументы, функция вернет количество присвоенных значений
<code>fpassthru(\$handle)</code>	Читает содержимое открытого файла <code>\$handle</code> от текущей позиции до конца файла и выводит в окно браузера
<code>file_get_contents(\$filename [, \$flags [, \$context [, \$offset [, \$maxlen]]]])</code>	Читает файл с именем <code>\$filename</code> в режиме <code>\$flags</code> и возвращает его содержимое в виде строки. Необязательный параметр <code>\$context</code> позволяет задать HTTP-заголовки, отправляемые сервером при обращении к сетевому файлу. Если это не требуется, вместо него можно передать <code>NULL</code> . Параметр <code>\$offset</code> задает позицию, начиная с которой выполняется чтение, а <code>\$maxlen</code> — количество символов, которые



	следует прочитать
<code>file(\$filename [, \$flags [, \$context]])</code>	Читает файл с именем <code>\$filename</code> и возвращает его содержимое в виде массива, каждый элемент которого соответствует отдельной строке. Необязательный параметр <code>\$flags</code> позволяет задать режим чтения файла. Параметр <code>\$context</code> позволяет задать HTTP-заголовки, отправляемые сервером при обращении к сетевому файлу
<code>readfile(\$filename [, \$use_include_path [, \$context]])</code>	Читает файл с именем <code>\$filename</code> и выводит его содержимое в окно браузера. Необязательный параметр <code>\$use_include_path</code> позволяет задать путь для поиска файла. Параметр <code>\$context</code> позволяет задать HTTP-заголовки, отправляемые сервером при обращении к сетевому файлу
<code>fwrite(\$handle, \$str [, \$length])</code>	Записывает в открытый файл <code>\$handle</code> содержимое строки <code>\$str</code> . Необязательный параметр <code>\$length</code> позволяет задать количество байтов, предназначенных для записи (по достижении этого количества запись останавливается). В случае успеха функция возвращает количество записанных байтов, в случае неудачи — <code>FALSE</code>
<code>fputs()</code>	Синоним для функции <code>fwrite()</code>
<code>file_put_contents(\$filename, \$data [, \$flags [, \$context]])</code>	Записывает в файл с именем <code>\$filename</code> данные из параметра <code>\$data</code> (либо строка, либо одномерный массив). Необязательный параметр <code>\$flags</code> определяет режим записи файла. Параметр <code>\$context</code> позволяет задать HTTP-заголовки, отправляемые сервером при обращении к сетевому файлу
<code>fflush(\$handle)</code>	Сбрасывает информацию, предназначенную для записи в файл из оперативной памяти на жесткий диск. Использование этой функции предотвращает потерю записанной в файл информации при нештатном завершении скрипта
<code>flock(\$handle, \$operation [, \$wouldblock])</code>	Устанавливает для открытого файла с дескриптором <code>\$handle</code> режим блокировки <code>\$operation</code> . Необязательный параметр <code>wouldblock</code> позволяет выяснить успешность операции
<code>ftruncate(\$handle, \$size)</code>	Уменьшает размер файла с дескриптором <code>\$handle</code> до <code>\$size</code> байтов. Возвращает <code>TRUE</code> в случае успеха и <code>FALSE</code> в противном случае

## Права доступа к файлу

Функция	Описание
chmod(\$filename, \$mode)	Устанавливает права доступа \$mode для файла \$filename. Возвращает TRUE в случае успеха и FALSE — в случае неудачи
chgrp(\$filename, \$group)	Устанавливает группу владельца \$group (имя группы или уникальный идентификатор GID) для файла \$filename. Возвращает TRUE в случае успеха и FALSE — в случае неудачи
chown(\$filename, \$user)	Устанавливает владельца \$user (имя владельца или уникальный идентификатор UID) для файла \$filename. Возвращает TRUE в случае успеха и FALSE — в случае неудачи
lchgrp(\$link, \$group)	Осуществляет попытку установить группу владельца \$group (имя группы или уникальный идентификатор GID) для символической ссылки \$link. Возвращает TRUE в случае успеха и FALSE — в случае неудачи
lchown(\$link, \$user)	Осуществляет попытку установить владельца \$user (имя владельца или уникальный идентификатор GID) для символической ссылки \$link. Возвращает TRUE в случае успеха и FALSE — в случае неудачи
fileperms(\$filename)	Возвращает права доступа файла \$filename. В случае неудачи возвращает FALSE
filegroup(\$filename)	Возвращает уникальный идентификатор группы владельца GID. В случае неудачи возвращает FALSE
fileowner(\$filename)	Возвращает уникальный идентификатор владельца UID файла \$filename. В случае неудачи возвращает FALSE
umask([\$mask])	Устанавливает права доступа по умолчанию для вновь создаваемых файлов и каталогов
is_readable(\$filename)	Возвращает TRUE, если файл \$filename доступен для чтения, в противном случае возвращает FALSE
is_writable(\$filename)	Возвращает TRUE, если файл \$filename доступен для записи, в противном случае возвращает FALSE
is_writeable()	Синоним функции is_writable()
is_executable(\$filename)	Возвращает TRUE, если файл \$filename доступен для выполнения, в противном случае возвращает FALSE

## Работа с каталогами

Функция	Описание
---------	----------

<code>mkdir(\$pathname [, \$mode [, \$recursive [, \$context]])</code>	Создает каталог <code>\$pathname</code> с правами доступа <code>\$mode</code> . Если необязательный параметр <code>\$recursive</code> принимает значение <code>TRUE</code> , все несуществующие каталоги в пути <code>\$pathname</code> создаются. Параметр <code>\$context</code> позволяет задать заголовки, отправляемые сервером при обращении к сетевому файлу. При успешном создании каталога функция возвращает <code>TRUE</code> , в противном случае — <code>FALSE</code>
<code>rmdir(\$dirname [, \$context])</code>	Удаляет пустой каталог с именем <code>\$dirname</code> . При успешном удалении каталога функция возвращает <code>TRUE</code> , в противном случае — <code>FALSE</code>
<code>getcwd()</code>	Возвращает полный путь к текущему каталогу
<code>chdir(\$directory)</code>	Изменяет текущий каталог на новый, указанный в параметре <code>\$directory</code> . При успешной смене текущего каталога функция возвращает <code>TRUE</code> , в противном случае — <code>FALSE</code>
<code>chroot(\$directory)</code>	Изменяет корневой каталог текущего процесса на <code>\$directory</code> . При успешной смене корневого каталога возвращает <code>TRUE</code> , в противном случае — <code>FALSE</code>
<code>opendir(\$path [, \$context])</code>	"Открывает" каталог — функция возвращает дескриптор, позволяющий читать содержимое каталога <code>\$path</code>
<code>readdir(\$dir_handle)</code>	Читает одну позицию из каталога и передвигает дескриптор на одну позицию вперед. Последовательный вызов функции позволяет прочитать содержимое каталога файл за файлом. Функция принимает в качестве единственного параметра <code>\$dir_handle</code> дескриптор, полученный ранее при помощи функции <code>opendir()</code> . Если достигнут конец каталога, функция возвращает <code>FALSE</code>
<code>closedir(\$dir_handle)</code>	Закрывает дескриптор <code>\$dir_handle</code> , открытый ранее при помощи функции <code>closedir()</code>
<code>rewinddir(\$dir_handle)</code>	Помещает дескриптор <code>\$dir_handle</code> в начало каталога
<code>scandir(\$directory [, \$sorting_order [, \$context]])</code>	Возвращает массив с содержимым каталога <code>\$directory</code> или <code>FALSE</code> , если <code>\$directory</code> не является каталогом. Содержимое каталога сортируется в алфавитном порядке, если необязательный параметр <code>\$sorting_order</code> принимает значение <code>TRUE</code> , элементы результирующего массива сортируются в обратном алфавитном порядке
<code>glob(\$pattern [, \$flags])</code>	Возвращает массив с именами файлов и подкаталогов, путь к которым удовлетворяет шаблону <code>\$pattern</code> . Необязательный параметр <code>\$flags</code> позволяет задать режим интерпретации шаблона