#### Работа с датами в PHP

## Функция time, формат timestamp

Функция **time** возвращает разницу в секундах между **1-го января 1970 года** и **текущим моментом времени**. Такое представление даты называется форматом **timestamp**.

## Зачем нужен timestamp?

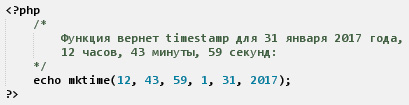
Время в формате **timestamp** используется для того, чтобы найти разницу между датами в секундах.

К примеру, есть timestamp за *1-го января текущего года* и timestamp *за текущий момент времени*. Если отнять один от другого, то получим разницу между этими моментами **в секундах**. Если поделить эту разницу на 60, то получим разницу в минутах (так как в минуте 60 секунд). Еще на 60 - и получим разницу в часах. И так далее.

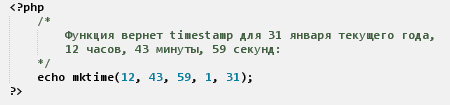
С помощью функции **time**, можно получить только текущий момент времени. Чтобы получить timestamp за **любую дату** следует использовать функцию **mktime**:

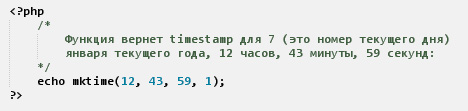
## Функция mktime

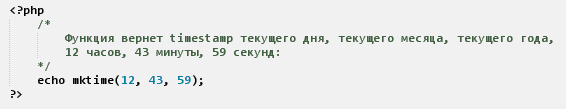
Функция **mktime** работает аналогично функции **time**, но, в отличие от нее, принимает параметры: **mktime(час, минута, секунда, месяц, день, год)** (обратите внимание на то, что месяц и день переставлены местами). Посмотрите примеры работы:



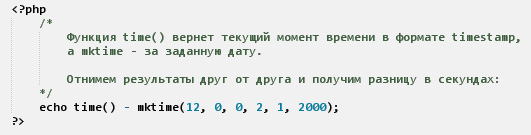
**Параметры можно опускать с конца**. Если мы не напишем год (последний параметр) - то автоматически возьмется текущий год, если не напишем еще и день (предпоследний параметр) - то возьмется текущий день и текущий год. И так далее:







Решим типичную **задачу** на **time** и **mktime**. Пусть нам нужно найти **разницу в секундах** между текущим моментом времени и полуднем (12 часов, 0 минут, 0 секунд) 1-го февраля (февраль - это месяц номер 2) 2000 года:



**Полученная разница в секундах** будет выглядеть так: **557992339** (обновите страницу и это число поменяется).

## Функция date

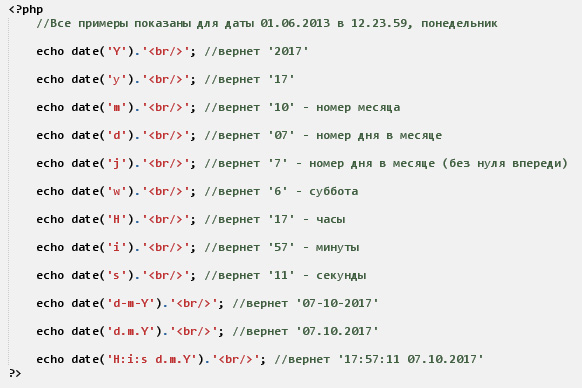
Функция **date** выводит текущие дату и время в заданном формате.

**Формат** задается управляющими командами (латиницей), при этом можно вставлять любые разделители между ними (дефисы, двоеточие и так далее).

**Команды** (заглавные буквы отличаются от обычных, обратите внимание!):

* **U** – количество секунд, прошедших с 1 января 1970 года (то есть timestamp).
* **z** – номер дня от начала года.
* **Y** – год, 4 цифры.
* **y** - год, две цифры.
* **m** – номер месяца (с нулем впереди).
* **n** – номер месяца без нуля впереди.
* **d** – номер дня в месяце, всегда две цифры (то есть первая может быть нулем).
* **j** – номер дня в месяце без предваряющего нуля.
* **w** – день недели (0 - воскресенье, 1 - понедельник и т.д.).
* **h** – часы в 12-часовом формате.
* **H** – часы в 24-часовом формате.
* **i** – минуты.
* **s** – секунды.
* **L** – 1, если високосный год, 0, если не високосный.
* **W** – порядковый номер недели года.
* **t** – количество дней в указанном месяце.

**Примеры работы с date:**



## Второй параметр функции date

Функция **date** имеет второй **необязательный параметр**, который принимает момент времени в формате timestamp. Если передать этот параметр, то функция date отформатирует не текущий момент времени, а тот, который передан вторым параметром. Этот **timestamp** можно получить, к примеру, через **mktime** (но не обязательно):

Безимени.jpg

Безимени-2.jpg

## Функция strtotime

Функция **strtotime** - это аналог функции mktime (тоже возвращает timestamp), только в отличие от нее принимает дату в **более** свободном формате.

К примеру, я могу передать ей строку **'2025-12-31'** и функция сама разберет, где тут год, где месяц, а где день, и вернет эту дату в формате timestamp.

Что можно делать еще: можно написать так - **strtotime('now')** - и мы получим текущий момент времени, или так - **strtotime('next Monday')** - и мы получим следующий понедельник (Monday по-английски 'понедельник').

Примеры работы:



## Как добавить или отнять дату

Чтобы отнять или прибавить к дате определенный промежуток времени, следует пользоваться комбинацией из трех функций: [**date\_create**](http://theory.phphtml.net/php/date/date_create.html)**,** [**date\_modify**](http://theory.phphtml.net/php/date/date_modify.html)**,** [**date\_format**](http://theory.phphtml.net/php/date/date_format.html).

Функция **date\_create** создает объект 'дата', с которым в дальнейшем можно выполнять некоторые операции.

date\_create([дата]);

Команда date\_create() возвращает текущий момент времени в объекте [DateTime](http://php.net/manual/ru/datetime.construct.php).

Функция **date\_modify** позволяет прибавлять и отнимать от даты определенные промежутки времени. Дата при этом должна быть объектом [DateTime](http://php.net/manual/ru/datetime.construct.php).

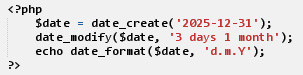
date\_modify(объект [DateTime](http://php.net/manual/ru/datetime.construct.php), что прибавить или отнять);

Функция изменяет сам переданный объект, и возвращает также измененный объект [DateTime](http://php.net/manual/ru/datetime.construct.php).

Функция **date\_format** выводит данные из объекта [DateTime](http://php.net/manual/ru/datetime.construct.php) в определенном формате. Управляющие команды для формата такие же, как в функции [date](http://theory.phphtml.net/php/date/date.html).

date\_create(объект [DateTime](http://php.net/manual/ru/datetime.construct.php), формат вывода);

Давайте создадим объект с датой за 2025 год, 12 месяц, 31 день, затем прибавим к ней 3 дня и 1 месяц и выведем в формате 'день.месяц.год':



Результат выполнения кода: 03.02.2026

# Регулярные выражения

Регулярные выражения - это строковые шаблоны. По этим шаблонам можно искать вхождения, производить замену, проверять на соответствие шаблону.

### Правила составление шаблона (pattern)

Границы шаблона должны обозначаться определенными символами ("**/**" или "**#**"):

/ТутТелоРегулярногоВыражения/ или #ТутТелоРегулярногоВыражения#.

Внутри регулярного выражения используются скобки - это подвыражения, с которыми можно производить манипуляции, пример:

**#^/catalog/([A-Za-z0-9\_\-\.]+)/([A-Za-z0-9\_\-\.]+)\.html.\*#**

Это выражение предназначено для получения параметров в строке **URL**. В начале строки идет спецсимвол "**^**" - это означает начало строки. Далее идет "**/catalog/**" - тут нет спецсимволов, это просто текст, который должен содержаться в строке. Затем встретили круглые скобки, т.е. дошли до первого подвыражения. В квадратных скобках обозначаются множество символов, которые могут быть в строке в этом месте. Знак "**-**" означает перечисление. Знак "**\**" экранирует специальные символы. Таким образом, в первом подвыражении у нас могут идти заглавные и прописные буквы латинского алфавита, цифры от 0 до 9, знак подчеркивания, тире и точка. Тире и точка - это спецсимволы, но здесь они экранированы, поэтому здесь это просто символы. После квадратных скобок идет знак "**+**" - это значит предыдущий символ (а у нас это множество символов заданное в квадратных скобках) может идти 1 или больше раз. Затем идет "**/**" - это просто символ, и аналогичное второе подвыражение. Затем идет "**\.html**", что означает текст "**.html**". А затем спец символы "**.\***" точка означает любой символ, а звездочка любое количество предыдущего символа. Т.е. после "**.html**" может идти все что угодно.

### Указание количества, квантификаторы

Выше мы уже рассмотрели такие символы, указывающие количество предыдущих символов, как **+** и **\***. Приведем все возможности указания количества:

|  |  |
| --- | --- |
| **{X}** | Х это число повторений, пример: **[0-9]{5}**, это значит что здесь должно быть 5 цифр |
| **{X,Y}** | X минимальное количество, Y - максимальное количество повторений. Пример **[0-9]{1,2}** - 1 или 2 цифры. **[0-9]{,2}** -0, 1 или 2 цифры. **[0-9]{2,}** - 2 и более цифр |
| **\*** | любое количество символов (включая 0 символов) |
| **+** | 1 и более символов |
| **?** | 1 или 0 символов |

### Спецсимволы

|  |  |
| --- | --- |
| **\** | экранирование спецсимвола |
| **\\** | обратный слеш |
| **|** | Метасимвол выбора (или или) |
| **^** | Метасимвол начала строки |
| **$** | Метасимвол конца строки |
| **\n** | Символ перевода строки (шестнадцатеричный код 0x0A) |
| **\r** | Символ возврата каретки (шестнадцатеричный код 0x0D) |
| **\t** | Символ табуляции (шестнадцатеричный код 0x09) |
| **.** | Точка. Любой символ. |

|  |  |
| --- | --- |
|  |  |

Для некоторых групп символов есть специальные сокращения:

|  |  |
| --- | --- |
| **\d** | Цифра (0-9), вместо **[0-9]** |
| **\D** | Не цифра (любой символ кроме символов **0-9**) |
| **\s** | Пустой символ (обычно пробел и символ табуляции) |
| **\S** | Непустой символ (все, кроме символов, определяемых метасимволом **\s**) |
| **\w** | "Словарный" символ (символ, который используется в словах. Обычно все буквы, все цифры и знак подчеркивания ('**\_**')) |
| **\W** | Все, кроме символов, определяемых метасимволом **\w** |

## Функции для работы с регулярными выражениями

### preg\_match

Выполняет проверку на соответствие регулярному выражению. При этом если совпадений будет много, то функция найдет **только первое** и закончит свою работу.

preg\_match (регулярное выражение, строка поиска);

//вернет:

//0 - соответствие не найдено

//1 - соответствие найдено

//false - произошла ошибка

Безимени.jpg

### preg\_match\_all

Выполняет глобальный поиск шаблона в строке. Возвращает количество найденных вхождений шаблона (которое может быть и нулем) либо FALSE, если во время выполнения возникли какие-либо ошибки.

$res = array();

preg\_match\_all (регулярное выражение, строка поиска, $res);

В переменную **$res** попадет результат поиска в виде массива: [0] содержит массив полных вхождений шаблона, элемент [1] содержит массив вхождений первой подмаски, и так далее.



### preg\_replace

Осуществляет поиск и замену:

preg\_replace(регулярное выражение, строка замещения, строка поиска);

Каждое значение может быть строкой или массивом строк, в случае, если строка поиска является массивом, то возвращается массив строк, иначе строка.