Calendar

Introduction

The calendar extension presents a series of functions to simplify converting between different calendar formats. The intermediary or standard it is based on is the Julian Day Count. The Julian Day Count is a count of days starting from January 1st, 4713 B.C. To convert between calendar systems, you must first convert to Julian Day Count, then to the calendar system of your choice. Julian Day Count is very different from the Julian Calendar! For more information on Julian Day Count, visit » http://www.hermetic.ch/cal_stud/jdn.htm. For more information on calendar systems visit » http://www.fourmilab.ch/documents/calendar/. Excerpts from this page are included in these instructions, and are in quotes.

Installing/Configuring

Requirements

No external libraries are needed to build this extension.

Installation

To get these functions to work, you have to compile PHP with *--enable-calendar*.

The Windows version of PHP has built-in support for this extension. You do not need to load any additional extensions in order to use these functions.

Runtime Configuration

This extension has no configuration directives defined in *php.ini*.

Resource Types

This extension has no resource types defined.

Predefined Constants

The constants below are defined by this extension, and will only be available when the extension has either been compiled into PHP or dynamically loaded at runtime.

CAL_GREGORIAN (integer)

CAL_JULIAN (integer)

CAL_JEWISH (integer)

CAL_FRENCH (integer)

CAL_NUM_CALS (integer)

CAL_DOW_DAYNO (integer)

CAL_DOW_SHORT (integer)

CAL_DOW_LONG (integer)

CAL_MONTH_GREGORIAN_SHORT (integer)

CAL_MONTH_GREGORIAN_LONG (integer)

CAL_MONTH_JULIAN_SHORT (integer)

CAL_MONTH_JULIAN_LONG (integer)

CAL_MONTH_JEWISH (integer)

CAL_MONTH_FRENCH (integer)

The following constants are available since PHP 4.3.0 :

CAL_EASTER_DEFAULT (integer)

CAL_EASTER_ROMAN (integer)

CAL_EASTER_ALWAYS_GREGORIAN (integer)

CAL_EASTER_ALWAYS_JULIAN (integer)

The following constants are available since PHP 5.0.0 :

CAL_JEWISH_ADD_ALAFIM_GERESH (integer)

CAL_JEWISH_ADD_ALAFIM (integer)

CAL_JEWISH_ADD_GERESHAYIM (integer)

Calendar Functions

cal_days_in_month

cal_days_in_month -- Return the number of days in a month for a given year and calendar

Description

int **cal_days_in_month** (int `$calendar`, int `$month`, int `$year`)

This function will return the number of days in the *`month`* of *`year`* for the specified *`calendar`*.

Parameters

`calendar`
: Calendar to use for calculation

`month`
: Month in the selected calendar

`year`
: Year in the selected calendar

Return Values

The length in days of the selected month in the given calendar

Examples

Example #1 - cal_days_in_month() example

```
<?php
$num = cal_days_in_month(CAL_GREGORIAN, 8, 2003); // 31
echo "There was $num days in August 2003";
?>
```

cal_from_jd

cal_from_jd -- Converts from Julian Day Count to a supported calendar

Description

array **cal_from_jd** (int `$jd`, int `$calendar`)

cal_from_jd() converts the Julian day given in *`jd`* into a date of the specified *`calendar`*. Supported *`calendar`* values are **CAL_GREGORIAN**, **CAL_JULIAN**, **CAL_JEWISH** and **CAL_FRENCH**.

Parameters

`jd`
Julian day as integer

`calendar`
Calendar to convert to

Return Values

Returns an array containing calendar information like month, day, year, day of week, abbreviated and full names of weekday and month and the date in string form "month/day/year".

Examples

Example #2 - cal_from_jd() example

```
<?php
$today = unixtojd(mktime(0, 0, 0, 8, 16, 2003));
print_r(cal_from_jd($today, CAL_GREGORIAN));
?>
```

The above example will output:

```
Array
(
   [date] => 8/16/2003
   [month] => 8
   [day] => 16
   [year] => 2003
   [dow] => 6
   [abbrevdayname] => Sat
   [dayname] => Saturday
   [abbrevmonth] => Aug
```

```
    [monthname] => August
)
```

See Also

- cal_to_jd()
- jdtofrench()
- jdtogregorian()
- jdtojewish()
- jdtojulian()
- jdtounix()

cal_info

cal_info -- Returns information about a particular calendar

Description

array **cal_info** ([int `$calendar`])

cal_info() returns information on the specified *`calendar`*.

Calendar information is returned as an array containing the elements *calname*, *calsymbol*, *month*, *abbrevmonth* and *maxdaysinmonth*. The names of the different calendars which can be used as *`calendar`* are as follows:

- 0 or **CAL_GREGORIAN** - Gregorian Calendar
- 1 or **CAL_JULIAN** - Julian Calendar
- 2 or **CAL_JEWISH** - Jewish Calendar
- 3 or **CAL_FRENCH** - French Revolutionary Calendar

If no *`calendar`* is specified information on all supported calendars is returned as an array.

Parameters

`calendar`
: Calendar to return information for. If no calendar is specified information about all calendars is returned.

Return Values

ChangeLog

Version	Description
Since 5.0	The *`calendar`* parameter becomes optional and defaults to "all calendars" if omitted.

Examples

Example #3 - cal_info() example

```
<?php
$info = cal_info(0);
print_r($info);
?>
```

The above example will output:

```
Array
(
   [months] => Array
       (
           [1] => January
           [2] => February
           [3] => March
           [4] => April
           [5] => May
           [6] => June
           [7] => July
           [8] => August
           [9] => September
           [10] => October
           [11] => November
           [12] => December
       )

   [abbrevmonths] => Array
       (
           [1] => Jan
           [2] => Feb
           [3] => Mar
           [4] => Apr
           [5] => May
           [6] => Jun
           [7] => Jul
           [8] => Aug
           [9] => Sep
           [10] => Oct
           [11] => Nov
           [12] => Dec
       )

   [maxdaysinmonth] => 31
   [calname] => Gregorian
   [calsymbol] => CAL_GREGORIAN
)
```

cal_to_jd

cal_to_jd -- Converts from a supported calendar to Julian Day Count

Description

int **cal_to_jd** (int `$calendar`, int `$month`, int `$day`, int `$year`)

cal_to_jd() calculates the Julian day count for a date in the specified *`calendar`*. Supported *`calendar`* s are **CAL_GREGORIAN**, **CAL_JULIAN**, **CAL_JEWISH** and **CAL_FRENCH**.

Parameters

`calendar`
: Calendar to convert from, one of **CAL_GREGORIAN**, **CAL_JULIAN**, **CAL_JEWISH** or **CAL_FRENCH**.

`month`
: The month as a number, the valid range depends on the *`calendar`*

`day`
: The day as a number, the valid range depends on the *`calendar`*

`year`
: The year as a number, the valid range depends on the *`calendar`*

Return Values

A Julian Day number.

See Also

- cal_from_jd()
- frenchtojd()
- gregoriantojd()
- jewishtojd()
- juliantojd()
- unixtojd()

easter_date

easter_date -- Get Unix timestamp for midnight on Easter of a given year

Description

int **easter_date** ([int `$year`])

Returns the Unix timestamp corresponding to midnight on Easter of the given year.

Warning
This function will generate a warning if the year is outside of the range for Unix timestamps (i.e. before 1970 or after 2037).

The date of Easter Day was defined by the Council of Nicaea in AD325 as the Sunday after the first full moon which falls on or after the Spring Equinox. The Equinox is assumed to always fall on 21st March, so the calculation reduces to determining the date of the full moon and the date of the following Sunday. The algorithm used here was introduced around the year 532 by Dionysius Exiguus. Under the Julian Calendar (for years before 1753) a simple 19-year cycle is used to track the phases of the Moon. Under the Gregorian Calendar (for years after 1753 - devised by Clavius and Lilius, and introduced by Pope Gregory XIII in October 1582, and into Britain and its then colonies in September 1752) two correction factors are added to make the cycle more accurate.

(The code is based on a C program by Simon Kershaw, <webmaster at ely.anglican dot org>)

Parameters

`year`
: The year as a number between 1970 an 2037

Return Values

The easter date as a unix timestamp.

ChangeLog

Version	Description
Since 4.3.0	The *`year`* parameter is optional and defaults

	to the current year according to the local time if omitted.

Examples

Example #4 - easter_date() example

```
<?php

echo date("M-d-Y", easter_date(1999));        // Apr-04-1999
echo date("M-d-Y", easter_date(2000));        // Apr-23-2000
echo date("M-d-Y", easter_date(2001));        // Apr-15-2001

?>
```

See Also

- easter_days() for calculating Easter before 1970 or after 2037

easter_days

easter_days -- Get number of days after March 21 on which Easter falls for a given year

Description

int **easter_days** ([int `$year` [, int `$method`]])

Returns the number of days after March 21 on which Easter falls for a given year. If no year is specified, the current year is assumed.

This function can be used instead of easter_date() to calculate Easter for years which fall outside the range of Unix timestamps (i.e. before 1970 or after 2037).

The date of Easter Day was defined by the Council of Nicaea in AD325 as the Sunday after the first full moon which falls on or after the Spring Equinox. The Equinox is assumed to always fall on 21st March, so the calculation reduces to determining the date of the full moon and the date of the following Sunday. The algorithm used here was introduced around the year 532 by Dionysius Exiguus. Under the Julian Calendar (for years before 1753) a simple 19-year cycle is used to track the phases of the Moon. Under the Gregorian Calendar (for years after 1753 - devised by Clavius and Lilius, and introduced by Pope Gregory XIII in October 1582, and into Britain and its then colonies in September 1752) two correction factors are added to make the cycle more accurate.

(The code is based on a C program by Simon Kershaw, <webmaster at ely.anglican dot org>)

Parameters

`year`
: The year as a positive number

`method`
: Allows to calculate easter dates based on the Gregorian calendar during the years 1582 - 1752 when set to **CAL_EASTER_ROMAN**. See the calendar constants for more valid constants.

Return Values

The number of days after March 21st that the Easter Sunday is in the given *`year`*.

ChangeLog

Version	Description

Since 4.3.0	The *year* parameter is optional and defaults to the current year according to the local time if omitted.
Since 4.3.0	The *method* parameter was introduced.

Examples

Example #5 - easter_days() example

```
<?php

echo easter_days(1999);        // 14, i.e. April 4
echo easter_days(1492);        // 32, i.e. April 22
echo easter_days(1913);        //  2, i.e. March 23

?>
```

See Also

- easter_date()

FrenchToJD

FrenchToJD -- Converts a date from the French Republican Calendar to a Julian Day Count

Description

int **frenchtojd** (int `$month`, int `$day`, int `$year`)

Converts a date from the French Republican Calendar to a Julian Day Count.

These routines only convert dates in years 1 through 14 (Gregorian dates 22 September 1792 through 22 September 1806). This more than covers the period when the calendar was in use.

Parameters

`month`
: The month as a number from 1 (for VendÃ©miaire) to 13 (for the period of 5-6 days at the end of each year)

`day`
: The day as a number from 1 to 30

`year`
: The year as a number between 1 and 14

Return Values

The julian day for the given french revolution date as an integer.

See Also

- jdtofrench()
- cal_to_jd()

GregorianToJD

GregorianToJD -- Converts a Gregorian date to Julian Day Count

Description

int **gregoriantojd** (int `$month`, int `$day`, int `$year`)

Valid Range for Gregorian Calendar 4714 B.C. to 9999 A.D.

Although this function can handle dates all the way back to 4714 B.C., such use may not be meaningful. The Gregorian calendar was not instituted until October 15, 1582 (or October 5, 1582 in the Julian calendar). Some countries did not accept it until much later. For example, Britain converted in 1752, The USSR in 1918 and Greece in 1923. Most European countries used the Julian calendar prior to the Gregorian.

Parameters

`month`
The month as a number from 1 (for January) to 12 (for December)

`day`
The day as a number from 1 to 31

`year`
The year as a number between -4714 and 9999

Return Values

The julian day for the given gregorian date as an integer.

Examples

Example #6 - Calendar functions

```
<?php
$jd = GregorianToJD(10, 11, 1970);
echo "$jd\n";
$gregorian = JDToGregorian($jd);
echo "$gregorian\n";
?>
```

See Also

- jdtogregorian()
- cal_to_jd()

JDDayOfWeek

JDDayOfWeek -- Returns the day of the week

Description

mixed **jddayofweek** (int `$julianday` [, int `$mode`])

Returns the day of the week. Can return a string or an integer depending on the mode.

Parameters

`julianday`
A julian day number as integer

`mode`

Calendar week modes

Mode	Meaning
0 (Default)	Return the day number as an int (0=Sunday, 1=Monday, etc)
1	Returns string containing the day of week (English-Gregorian)
2	Return a string containing the abbreviated day of week (English-Gregorian)

Return Values

The gregorian weekday as either an integer or string.

JDMonthName

JDMonthName -- Returns a month name

Description

string **jdmonthname** (int `$julianday`, int `$mode`)

Returns a string containing a month name. *`mode`* tells this function which calendar to convert the Julian Day Count to, and what type of month names are to be returned.

Calendar modes

Mode	Meaning	Values
0	Gregorian - abbreviated	Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec
1	Gregorian	January, February, March, April, May, June, July, August, September, October, November, December
2	Julian - abbreviated	Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec
3	Julian	January, February, March, April, May, June, July, August, September, October, November, December
4	Jewish	Tishri, Heshvan, Kislev, Tevet, Shevat, AdarI, AdarII, Nisan, Iyyar, Sivan, Tammuz, Av, Elul
5	French Republican	Vendemiaire, Brumaire, Frimaire, Nivose, Pluviose, Ventose, Germinal, Floreal, Prairial, Messidor, Thermidor, Fructidor, Extra

Parameters

jday
: The Julian Day to operate on

calendar
: The calendar to take the month name from

Return Values

The month name for the given Julian Day and *calendar*.

JDToFrench

JDToFrench -- Converts a Julian Day Count to the French Republican Calendar

Description

string **jdtofrench** (int `$juliandaycount`)

Converts a Julian Day Count to the French Republican Calendar.

Parameters

`julianday`
: A julian day number as integer

Return Values

The french revolution date as a string in the form "month/day/year"

See Also

- frenchtojd()
- cal_from_jd()

JDToGregorian

JDToGregorian -- Converts Julian Day Count to Gregorian date

Description

string **jdtogregorian** (int `$julianday`)

Converts Julian Day Count to a string containing the Gregorian date in the format of "month/day/year".

Parameters

`julianday`
: A julian day number as integer

Return Values

The gregorian date as a string in the form "month/day/year"

See Also

- gregoriantojd()
- cal_from_jd()

jdtojewish

jdtojewish -- Converts a Julian day count to a Jewish calendar date

Description

string **jdtojewish** (int `$juliandaycount` [, bool `$hebrew` [, int `$fl`]])

Converts a Julian Day Count to the Jewish Calendar.

Parameters

`julianday`
: A julian day number as integer

`hebrew`
: If the *`hebrew`* parameter is set to **TRUE**, the *`fl`* parameter is used for Hebrew, string based, output format.

`fl`
: The available formats are: **CAL_JEWISH_ADD_ALAFIM_GERESH**, **CAL_JEWISH_ADD_ALAFIM**, **CAL_JEWISH_ADD_GERESHAYIM**.

Return Values

The jewish date as a string in the form "month/day/year"

ChangeLog

Version	Description
5.0.0	The *`hebrew`* and *`fl`* parameters were added

Examples

Example #7 - jdtojewish() Example

```
<?php
echo jdtojewish(gregoriantojd(10, 8, 2002), true,
      CAL_JEWISH_ADD_GERESHAYIM + CAL_JEWISH_ADD_ALAFIM +
CAL_JEWISH_ADD_ALAFIM_GERESH);
```

```
?>
```

See Also

- jewishtojd()
- cal_from_jd()

JDToJulian

JDToJulian -- Converts a Julian Day Count to a Julian Calendar Date

Description

string **jdtojulian** (int `$julianday`)

Converts Julian Day Count to a string containing the Julian Calendar Date in the format of "month/day/year".

Parameters

`julianday`
: A julian day number as integer

Return Values

The julian date as a string in the form "month/day/year"

See Also

- juliantojd()
- cal_from_jd()

jdtounix

jdtounix -- Convert Julian Day to Unix timestamp

Description

int **jdtounix** (int `$jday`)

This function will return a Unix timestamp corresponding to the Julian Day given in *`jday`* or **FALSE** if *`jday`* is not inside the Unix epoch (Gregorian years between 1970 and 2037 or 2440588 <= *`jday`* <= 2465342). The time returned is localtime (and not GMT).

Parameters

`jday`
: A julian day number between 2440588 and 2465342.

Return Values

The unix timestamp for the start of the given julian day.

See Also

- unixtojd()

JewishToJD

JewishToJD -- Converts a date in the Jewish Calendar to Julian Day Count

Description

int **jewishtojd** (int `$month`, int `$day`, int `$year`)

Although this function can handle dates all the way back to the year 1 (3761 B.C.), such use may not be meaningful. The Jewish calendar has been in use for several thousand years, but in the early days there was no formula to determine the start of a month. A new month was started when the new moon was first observed.

Parameters

`month`
: The month as a number from 1 to 13

`day`
: The day as a number from 1 to 30

`year`
: The year as a number between 1 and 9999

Return Values

The julian day for the given jewish date as an integer.

See Also

- jdtojewish()
- cal_to_jd()

JulianToJD

JulianToJD -- Converts a Julian Calendar date to Julian Day Count

Description

int **juliantojd** (int `$month`, int `$day`, int `$year`)

Valid Range for Julian Calendar 4713 B.C. to 9999 A.D.

Although this function can handle dates all the way back to 4713 B.C., such use may not be meaningful. The calendar was created in 46 B.C., but the details did not stabilize until at least 8 A.D., and perhaps as late at the 4th century. Also, the beginning of a year varied from one culture to another - not all accepted January as the first month.

Caution
Remember, the current calendar system being used worldwide is the Gregorian calendar. gregoriantojd() can be used to convert such dates to their Julian Day count.

Parameters

`month`
: The month as a number from 1 (for January) to 12 (for December)

`day`
: The day as a number from 1 to 31

`year`
: The year as a number between -4713 and 9999

Return Values

The julian day for the given julian date as an integer.

See Also

- jdtojulian()
- cal_to_jd()

unixtojd

unixtojd -- Convert Unix timestamp to Julian Day

Description

int **unixtojd** ([int `$timestamp`])

Return the Julian Day for a Unix *`timestamp`* (seconds since 1.1.1970), or for the current day if no *`timestamp`* is given.

Parameters

`timestamp`
: A unix timestamp to convert.

Return Values

A julian day number as integer.

See Also

- jdtounix()