

# iranphp articles

عنوان مقاله : Some Date-Related functions  
نگارنده : محمد خالدي  
آدرس پست الكترونيك : khaaledy@ferdowsi.um.ac.ir  
تاريخ نگارش : .....

## :Some Date-Related functions

```
<?
/*
Module name: DateUtil.inc

Copyright (c) 2002 Muhammad Khaaleedy (khaaleedy@ferdowsi.um.ac.ir, http://www.um.ac.ir/~khaaleedy)
Last Modified: 02-20-2002

Hint:
All functions are coded based on the fact that, for every integer N,  $22 + [33 * N / 8]$  will yield a
Solar leap year. Here by [x] I mean, the maximum integer not greater than x.
The C version is also available. If you want it, just email me!

License:
This program is free software; you can redistribute it and/or modify it under the terms of the GNU
General Public License as published by the Free Software Foundation; either version 2 of the
License,
or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without
even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

If you don't have a copy of the GNU General Public License, write to the Free Software Foundation,
Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
*/
define("BASE_DAY", 5);
define("START_YEAR", 1);
define("BASE_YEAR", 1370);
/*
GetNOOfYear: Gets a Solar Year, and returns the corresponding N
*/
function GetNOOfYear($Year) {
    return round((8 * ($Year - 22)) / 33 + ($Year >= 22 ? 0.5 : -0.5));
}
/*
GetLeapYear: Gets an N, and returns the corresponding Solar Leap year
*/
function GetLeapYear($N) {
    return round(22 + 33 * $N / 8);
}
/*
IsLeap: Checks whether a Solar Year is leap or not
*/
function IsLeap($Year) {
    return $Year == GetLeapYear(GetNOOfYear($Year));
}
/*
IsSpecialLeap: As you may know, after 7 4-year-leap we have one 5-year-leap (hence, SPECIAL
LEAP).
This function, checks whether the Solar Year is a special leap or not
*/
function IsSpecialLeap($Year) {
    return ($Year - 22) % 33 == 0;
}
/*
IsChristianLeap: Checks whether the Christian Year is leap or not
*/
```

```
function IsChristianLeap($Year) {
    return (!(($Year % 4) && $Year % 100 || !($Year % 400));
}
/*
NearestLeapYear: Calculates the smallest Solar leap year greater than a given Solar Year
*/
function NearestLeapYear($Year) {
    while(!IsLeap($Year++));
    return $Year;
}
/*
FirstDayOfYear: Calculates the first day of a Solar Year (0 = Saturday, 1 = Sunday, 2 = Monday
etc.)
*/
function FirstDayOfYear($Year) {
    return ((BASE_DAY +
        $Year - BASE_YEAR + // Years Between
        GetNOOfYear(NearestLeapYear($Year)) -
        GetNOOfYear(NearestLeapYear(BASE_YEAR)) // Leaps Between
    ) % 7 + 7) % 7;
}
/*
PassedDaysOfYear: Calculates the number of passed days from the start of a Solar year
*/
function PassedDaysOfYear($Day, $Month) {
    return ($Month - 1) * 30 + ($Month < 7 ? $Month - 1 : 6) + $Day;
}
/*
DayOfWeek: Calculates the day of week of a given date (year, month, and day)
*/
function DayOfWeek($Year, $Month, $Day) {
    return (FirstDayOfYear($Year) + PassedDaysOfYear($Day, $Month) - 1) % 7;
}
/*
LeapYearsBetween: Calculates the number of leap years between 2 given Solar Years
*/
function LeapYearsBetween($FirstYear, $SecondYear) {
    return abs(GetNOOfYear(NearestLeapYear($FirstYear)) -
GetNOOfYear(NearestLeapYear($SecondYear)));
}
/*
IncrementDate: Increments a Solar date (year, month, and day), by one
*/
function IncrementDate($Day, $Month, $Year) {
    if(++$Day <= 29)
        return array($Day, $Month, $Year);
    switch($Day) {
        case 30:
            if($Month != 12 || IsLeap($Year))
                break;
            $Day = $Month = 1;
            $Year++;
            break;
        case 31:
            if($Month < 7)
                break;
            $Day = 1;
            if($Month < 12) {
                $Month++;
                break;
            }
    }
```

```
    }
    $Month = 1;
    $Year++;
    break;
case 32:
    $Day = 1;
    $Month++;
    break;
}
return array($Day, $Month, $Year);
}
/*
DecrementDate: Decrements a Solar date (year, month, and day), by one
*/
function DecrementDate($Day, $Month, $Year) {
    if(--$Day >= 1)
        return array($Day, $Month, $Year);
    if($Month == 1) {
        $Month = 12;
        $Day = 29 + IsLeap(--$Year);
    }
    else {
        $Month--;
        $Day = 31 - ($Month > 7);
    }
    return array($Day, $Month, $Year);
}
/*
DateBackward: Calculates the date of 'Gap' days BEFORE a given Solar date
*/
function DateBackward($Day, $Month, $Year, $Gap) {
    for($i = 0; $i < $Gap; ++$i)
        list($Day, $Month, $Year) = DecrementDate($Day, $Month, $Year);
    return array($Day, $Month, $Year);
}
/*
DateForward: Calculates the date of 'Gap' days AFTER a given Solar date
*/
function DateForward($Day, $Month, $Year, $Gap) {
    for($i = 0; $i < $Gap; ++$i)
        list($Day, $Month, $Year) = IncrementDate($Day, $Month, $Year);
    return array($Day, $Month, $Year);
}
/*
DaysPassedFromTheFirst: Calculates the number of days passed from the start of a Solar Date
*/
function DaysPassedFromTheFirst($Day, $Month, $Year) {
    return PassedDaysOfYear($Day, $Month) + ($Year - 1) * 365 + LeapYearsBetween($Year,
START_YEAR);
}
/*
DifferenceDate: Calculates the difference between two given Solar dates
*/
function DifferenceDate($Day1, $Month1, $Year1, $Day2, $Month2, $Year2) {
    return DaysPassedFromTheFirst($Day1, $Month1, $Year1) - DaysPassedFromTheFirst($Day2,
$Month2, $Year2);
}
/*
XPassedDaysOf: Calculates the number of passed days from the start of a Christian year
*/
```

```
function XPassedDaysOf($Day, $Month, $Year) {
    $XMonths = array(31, 28 + IsChristianLeap($Year), 31, 30, 31, 30, 31, 31, 30, 31, 30, 31);
    for($Days = $Cnt = 0; $Cnt < $Month - 1; $Cnt++)
        $Days += $XMonths[$Cnt];
    return $Days + $Day;
}
/*
ConvertX2SDate: Converts a Christian Date to the corresponding Solar one
*/
function ConvertX2SDate($Day, $Month, $Year) {
    $XDaysPassed = XPassedDaysOf($Day, $Month, $Year);
    $Year -= 622;
    $Month = 10;
    $Day = 11 + IsLeap($Year);
    return DateForward($Day, $Month, $Year, $XDaysPassed - 1);
}
/*
ConvertS2XDate: Converts a Solar Date to the corresponding Christian one
*/
function ConvertS2XDate($Day, $Month, $Year) {
    $today = date("Ymd");
    $yy = substr($today, 0, 4);
    $mm = substr($today, 4, 2);
    $dd = substr($today, 6, 2);
    list($dd, $mm, $yy) = ConvertX2SDate($dd, $mm, $yy); // Today's Solar date
    $Diff = DifferenceDate($Day, $Month, $Year, $dd, $mm, $yy);
    return date("Ymd", mktime(0, 0, 0, date("m"), date("d") + $Diff, date("Y")));
}
?>
```

Dear friends,

I wish here were a way to make me able to UPLOAD the source code for you. This is because pasting the code to the above textarea, will make it somewhat NASTY! :)

Ok, hopefully the code would be useful. I will be very pleased if you give me some feedbacks on it!

Regards,

Muhammad Khaaledy