The DateInterval class ¶

php.net/manual/en/class.dateinterval.php

Edit Report a Bug
(PHP 5 >= 5.3.0, PHP 7)

Introduction ¶

Represents a date interval.

A date interval stores either a fixed amount of time (in years, months, days, hours etc) or a relative time string in the format that DateTime's constructor supports.

Class synopsis ¶

DateInterval {

public integer *$y* ;
public integer *$m* ;
public integer *$d* ;
public integer *$h* ;
public integer *$i* ;
public integer *$s* ;
public float *$f* ;
public integer *$invert* ;
public mixed *$days* ;
public __construct (string `$interval_spec`)
public static DateInterval createFromDateString (string `$time`)
public string format (string `$format`)
}

Properties ¶

y
Number of years.

m
Number of months.

d
Number of days.

h
Number of hours.

i

Number of minutes.

s
Number of seconds.

f
Number of microseconds, as a fraction of a second.

invert
Is *1* if the interval represents a negative time period and *0* otherwise. See DateInterval::format().

days
If the DateInterval object was created by DateTime::diff(), then this is the total number of days between the start and end dates. Otherwise, *days* will be `FALSE`.

Before PHP 5.4.20/5.5.4 instead of `FALSE` you will receive -99999 upon accessing the property.

Changelog ¶

Version	Description
7.1.0	The *f* property was added.

Table of Contents ¶

add a note

User Contributed Notes 11 notes

up
down
84

php at keith tyler dot com ¶
6 years ago

`DateInterval does not support split seconds (microseconds or milliseconds etc.) when doing a diff between two DateTime objects that contain microseconds.`
`So you cannot do the following, for example:`

```
<?php
$d1=new DateTime("2012-07-08 11:14:15.638276");
$d2=new DateTime("2012-07-08 11:14:15.889342");
$diff=$d2->diff($d1);
print_r( $diff ) ;
```

```
DateInterval Object
(
    [y] => 0
    [m] => 0
    [d] => 0
    [h] => 0
    [i] => 0
    [s] => 0
    [invert] => 0
    [days] => 0
)

*/
?>
```

You get back 0 when you actually want to get 0.251066 seconds.

up
down
9

trap-phpdocs-06022017 at skvorc dot me ¶
1 year ago

IRT note below: from PHP 7.0+, split seconds get returned, too, and will be under the `f` property.

up
down
34

p dot scheit at ps-webforge dot com ¶
7 years ago

If you want to convert a Timespan given in Seconds into an DateInterval Object you could dot the following:

```
<?php

$dv = new DateInterval('PT'.$timespan.'S');
?>
```

but wenn you look at the object, only the $dv->s property is set.

As stated in the documentation to DateInterval::format

The DateInterval::format() method does not recalculate carry over points in time strings nor in date segments. This is expected because it is not possible to overflow values like "32 days" which could be interpreted as anything from "1 month and 4 days" to "1 month and 1 day".

If you still want to calculate the seconds into hours / days / years, etc do the following:

```
<?php
```

```
$d1 = new DateTime();
$d2 = new DateTime();
$d2->add(new DateInterval('PT'.$timespan.'S'));

$iv = $d2->diff($d1);

?>

$iv is an DateInterval set with days, years, hours, seconds, etc ...
```

up
down
6

0bccbf3a at opayq dot com ¶
4 years ago

```
invert flag is unreliable.
If you've created interval with \DateInterval::createFromDateString with
value like '1 day ago' than actually days counter will be negative, and
invert flag will be 0. Also, setting invert to 1 with negative units is not
working.
Reliable solution to check if interval is negative is to actually apply it
and compare:
<?php
    private function isNegative(\DateInterval $interval)
    {
        $now = new \DateTimeImmutable();
        $newTime = $now->add($interval);
        return $newTime < $now;
    }
?>
Also, if you want to compare some units of two intervals you should take
abs() of them. Or make whole interval absolute:
<?php
    private function absInterval(\DateInterval $interval)
    {
        $now = new \DateTimeImmutable();
        $new = $now->add($interval);
        $newInt = $now->diff($new);
        if (1 === $newInt->invert) {
            $newInt->invert = 0;
        }
        return $newInt;
    }
?>
P.S.: tested on 5.5.12-dev and 5.5.9
```

up
down
1

cmygind@tsccorp dot com ¶

3 years ago

```
You can create a series of dates starting with the first day of the week for each week, if you wish to populate list box on your web page with this date math. Use the absolute abs( ) function to convert negative numbers generated from dates in the past.
<?php
            $TwoWeeksAgo  = new DateTime(date("Ymd"));
             $TwoWeeksAgo->sub(new DateInterval('P'.abs ( (7-date("N")-14)).'D'));
             $LastWeek = new DateTime(date("Ymd"));
             $LastWeek->sub(new DateInterval('P'.abs ( (7-date("N")-7)).'D'));
             $ThisWeek = new DateTime(date("Ymd"));
             $ThisWeek->add(new DateInterval('P'.abs ( (7-date("N"))).'D'));
              echo  'Start of This week is '.$ThisWeek->format('l m/d/Y').'<br/>';
            echo  'Start of Last week is '.$LastWeek->format('l m/d/Y').'<br/>';
            echo  'Start of 2 weeks ago is '.$TwosWeekAgo->format('l m/d/Y').'<br/>';
?>
```

up
down
1

Miller ¶
5 years ago

```
This DateInterval extension allows you to write a formatted timestamp but omit the "zero values" and handle things like listing, plurals, etc.
Example input: '%y year(s)', '%m month(s)', '%d day(s)', '%h hour(s)', '%i minute(s)', '%s second(s)'
Example output: 1 year, 2 months, 16 days, 1 minute, and 15 seconds
Example input: '%y año(s)', '%m mes(es)', '%d día(s)', '%h hora(s)', '%i minuto(s)', '%s segundo(s)'
Example output: 1 año, 2 meses, 16 días, 1 minuto, y 15 segundos
<meta charset="UTF-8">
<?php
error_reporting(E_ALL);

class MyDateInterval extends DateInterval {
    public
        $pluralCheck = '()',
            $singularReplacement = '',
            $separator = ', ',
            $finalSeparator = ', and ',
            $finalSeparator2 = ' and ';
```

```
    public static function createFromDateInterval (DateInterval $interval) {
        $obj = new self('PT0S');
        foreach ( $interval as $property => $value) {
            $obj->$property = $value;
        }
        return $obj;
    }

    public function formatWithoutZeroes () {
        $parts = array ();
        foreach ( func_get_args() as $arg) {
            $pre = mb_substr($arg, 0, mb_strpos($arg, '%'));
            $param = mb_substr($arg, mb_strpos($arg, '%'), 2);
            $post = mb_substr($arg, mb_strpos($arg,
$param)+mb_strlen($param));
            $num = intval(parent::format($param));

            $open = preg_quote($this->pluralCheck[0], '/');
            $close = preg_quote($this->pluralCheck[1], '/');
            $pattern = "/$open(.*)$close/";
            list ( $pre, $post) = preg_replace($pattern, $num == 1 ? $this-
>singularReplacement : '$1', array ($pre, $post));

            if ( $num != 0) {
                $parts[] = $pre.$num.$post;
            }
        }

        $output = '';
        $l = count($parts);
        foreach ( $parts as $i => $part) {
            $output .= $part.($i < $l-2 ? $this->separator : ($l == 2 ?
$this->finalSeparator2 : ($i == $l-2 ? $this->finalSeparator : '')));
        }
        return $output;
    }
}

date_default_timezone_set('America/Phoenix');

$today = new DateTime('today');
echo 'Today is ', $today->format('F d, Y h:ia'), '.<br>', PHP_EOL;
    $expiration = new DateTime('today +1 year +2 months +16 days +1 minute
+15 seconds');
echo 'Expires ', $expiration->format('F d, Y h:ia'), '.<br>', PHP_EOL;


$interval = MyDateInterval::createFromDateInterval($today-
>diff($expiration));

echo 'That is ', $interval->formatWithoutZeroes('%y year(s)', '%m month(s)',
'%d day(s)', '%h hour(s)', '%i minute(s)', '%s second(s)'), ' from now.<br>',
PHP_EOL;
```

```
$interval->finalSeparator = ', y ';
$interval->finalSeparator2 = ' y ';
echo 'Que es de ', $interval->formatWithoutZeroes('%y año(s)', '%m mes(es)',
'%d día(s)', '%h hora(s)', '%i minuto(s)', '%s segundo(s)'), ' a partir de
ahora.';
    ?>
```

up
down
0

***theDustin* ¶**
5 years ago

If you want to format a \DateInterval to something like you input (new \DateInterval("P3W2D")) you can use one of this:

```
<?php

class MyDateInterval extends \DateInterval
{

    const INTERVAL_ISO8601 = 'P%yY%mM%dDT%hH%iM%sS';

    function __toString()
    {
        $sReturn = 'P';

        if($this->y){
            $sReturn .= $this->y . 'Y';
        }

        if($this->m){
            $sReturn .= $this->m . 'M';
        }

        if($this->d){
            $sReturn .= $this->d . 'D';
        }

        if($this->h || $this->i || $this->s){
            $sReturn .= 'T';

            if($this->h){
                $sReturn .= $this->h . 'H';
            }

            if($this->i){
                $sReturn .= $this->i . 'M';
            }

            if($this->s){
                $sReturn .= $this->s . 'S';
            }
        }
```

```
        return $sReturn;
    }
}

?>

example use:

<?php

$oDateIntervalValue = new MyDateInterval('P3M');

$sFormatResult = $oDateIntervalValue-
>format(MyDateInterval::INTERVAL_ICALENDAR); $sToStringResult = (string)
$oDateIntervalValue;

var_dump(new MyDateInterval($sFormatResult)); var_dump(new
MyDateInterval($sToStringResult));

?>
```

up
down
0

***computrius* ¶**
5 years ago

```
It appears that they "days" property that is populated by \DateTime::diff
does not contain a float for the differences in time.
It is rounded down to the nearest whole day.
    $d1 = new \DateTime("2013-07-31 10:29:00");
    $d2 = new \DateTime("2013-08-02 5:32:12");
    echo $d1->diff($d2)->days;

Output: 1
```

up
down
0

***artur at qrupa dot com* ¶**
5 years ago

```
When using DateInterval('P3M') on 30th of November you get March instead of
Ferbuary.
```

up
down
-2

***Joan* ¶**
4 years ago

```
I had the doubt after reading this page on how to create negative intervals.
```

So far the only solution is to create the interval and negativize it.

```
<?php
$date1 = new DateTime();
$eightynine_days_ago = new DateInterval( "P89D" );
$eightynine_days_ago->invert = 1; $date1->add( $eightynine_days_ago );
?>
```

and then $date1 is now 89 days in the past.

This information is extracted from another php comment http://www.php.net/manual/en/dateinterval.construct.php#102976 but this page seems to be the first place where people will look for it.

up
down
-15

till at php dot net ¶
5 years ago

It should be noted that the following code will not throw an exception or return false, or anything:

```
<?php
$interval = new \DateInterval::createFromDateString("this is not a date interval");
?>
```

Your best way to check if what you created is a "valid" interval, by doing something like the following:

```
<?php
$interval = new \DateInterval::createFromDateString("this is not a date interval");
if (0 == $interval->format('s')) {
     throw new \LogicException("Wrong interval");
}
?>
```