NAME

Date::Manip::Examples − examples of how to use Date::Manip

DESCRIPTION

This document includes a number of examples on how to do common Date::Manip operations. I will be happy to add new examples over time, and welcome suggestions and examples to include.

In most cases, an example will include two different ways of getting the answer. The first way will be using the new (as of 6.00) OO modules. The second will be using the old-style functional interface.

It should be noted that any time you want to work with alternate time zones, the OO interface is STRONGLY recommended since the functional interface does not preserve time zone information with the date, and may therefore give incorrect results in some cases. However, working in the time zone of the system should give correct results.

It should be noted that, in the examples below, it appears that the OO method often requires more lines of code than the functional interface. There are a number of ways to shorten the OO method, but for the examples, I wanted to include all the steps explicitly.

PARSING A DATE

Dates can be parsed in practically any form in common usage:

OO method

```
$date = new Date::Manip::Date;
$err = $date->parse("today");
$err = $date->parse("1st Thursday in June 1992");
$err = $date->parse("05/10/93");
$err = $date->parse("12:30 Dec 12th 1880");
$err = $date->parse("8:00pm December tenth");
```

Functional

```
$date = ParseDate("today");
$date = ParseDate("1st Thursday in June 1992");
$date = ParseDate("05/10/93");
$date = ParseDate("12:30 Dec 12th 1880");
$date = ParseDate("8:00pm December tenth");
```

The Date::Manip::Date manual has a list of all valid formats.

PARSING AN AMOUNT OF TIME

Amounts of time (referred to as deltas) can also be parsed:

OO method

```
$delta = new Date::Manip::Delta;
$err = $delta->parse("in 12 hours");
$err = $delta->parse("-1:30:0");
$err = $delta->parse("4 business days later");
```

Functional

```
$delta = ParseDateDelta("in 12 hours");
$delta = ParseDateDelta("-1:30:0");
$delta = ParseDateDelta("4 business days later");
```

TO CALCULATE THE AMOUNT OF TIME BETWEEN TWO DATES

```
$datestr1 = "Jan 30 1999 13:00 EST";
$datestr2 = "2/Mar/1999 15:30:00 +0500";
```

OO method

```
$date1 = new Date::Manip::Date;
$date2 = $date1->new_date();
$err = $date1->parse($datestr1);
$err = $date2->parse($datestr2);
```

To get an exact amount of time between the two dates (expressed only in terms of hours, minutes, seconds), use:

```
$delta = $date1->calc($date2);
```

To get an approximate amount of time (expressed in terms of years, months, weeks, etc. in terms that a human would typically think of), use:

```
$delta = $date1->calc($date2,"approx");
```

Functional

```
$date1 = ParseDate($string1);
$date2 = ParseDate($string2);
```

To get an exact amount:

```
$delta = DateCalc($date1,$date2);
```

and the approximate amount:

```
$delta = DateCalc($date1,$date2,1);
```

The Date::Manip::Calc manual has information about these, and other types of calculations.

TO ADD AN AMOUNT OF TIME TO A DATE

To find a second date a given amount of time before or after a first date, use the following:

```
$datestr  = "Jan 30 1999 13:00 EST";
$deltastr = "12 hours ago";
$deltastr = "in 3 business days";
```

OO method

```
$date = new Date::Manip::Date;
$delta = $date->new_delta();
$date->parse($datestr);
$delta->parse($deltastr);

$d = $date->calc($delta);
```

Functional

```
$date = DateCalc($datestr,$deltastr);
```

If the delta is a business delta, it will do a business mode calculation.

The Date::Manip::Calc manual has information about these, and other types of calculations.

COMPARE TWO DATES

To take two different dates and see which is earlier, do the following:

```
$datestr1 = "Jan 30 1999 13:00 EST";
$datestr2 = "2/Mar/1999 15:30:00 +0500";
```

OO method

```
$date1 = new Date::Manip::Date;
$date2 = $date1->new_date;
$date1->parse($datestr1);
$date2->parse($datestr2);

$date1->cmp($date2);
   => -1, 0, 1
```

Functional

```
$date1 = ParseDate($datestr1);
$date2 = ParseDate($datestr2);

Date_Cmp($date1,$date2);
   => −1, 0, 1
```

TO EXTRACT INFORMATION ABOUT A DATE OR DELTA

If you have a date or a delta, you can extract information about them as follows:

```
$datestr  = "1:24:08 PM EST Feb 3, 1996";
$deltastr = "12 hours ago";
```

OO method

```
$date = new Date::Manip::Date;
$delta = $date−>new_delta();
$date−>parse($datestr);
$delta−>parse($deltastr);

$str = $date−>printf("It is now %T on %b %e, %Y.");
  =>  "It is now 13:24:08 on Feb  3, 1996."

$str = $delta−>printf("In %hv hours, %mv minutes, %sv seconds");
  => "In −12 hours, 0 minutes, 0 seconds";
```

Functional

```
$str = UnixDate($datestr,"It is now %T on %b %e, %Y.");
  =>  "It is now 13:24:08 on Feb  3, 1996."

$str = Delta_Format($deltastr,"In %hv hours, %mv minutes, %sv seconds");
  => "In −12 hours, 0 minutes, 0 seconds";
```

The Date::Manip::Date manual contains all of the format codes that can be used to extract information from a date. The Date::Manip::Delta manual contains the codes for a delta.

WORKING WITH EPOCH

Date::Manip can easily be used to work with the number of seconds since the epoch (Jan 1, 1970 00:00:00 UTC).

If you have a date, and you want to find out how many seconds it is after the epoch, you can do it in the following ways:

```
$datestr  = "1999−04−30−15:30:00 EDT";
$secs  = 1234567;
```

OO method

To find out how many seconds have elapsed on a certain date, you can do the following:

```
$date     = new Date::Manip::Date;
$err      = $date−>parse($datestr);

$str      = $date−>printf('%s');
   => number of seconds
```

To find out the date that is a certain number of seconds since the epoch, you can use the following:

```
$date  = new Date::Manip::Date;
$err   = $date−>parse("epoch $secs");
```

`$date` now contains the date wanted (in the local time zone)

Functional

To find out how many seconds have elapsed:

```
$str = UnixDate($datestr,'%s');
  => number of seconds
```

To find the date that is a number of seconds since the epoch:

```
$date = ParseDateString("epoch $secs");
```

Note that Date::Manip will work with both positive seconds (for dates that have come since the epoch) and negative seconds (for dates that occurred before the epoch).

RECURRING EVENTS

To find a list of dates where a recurring event happens (even very complex recurrences), do the following:

OO method

```
# To find the 2nd Tuesday of every month from Jan 1 1999 to Apr 30 1999

$recur = new Date::Manip::Recur;
$start = $recur->new_date();
$end   = $recur->new_date();
$start->parse("Jan 1 1999");
$end->parse("Apr 30 1999");

$recur->parse("0:1*2:2:0:0:0",$start,$end);
@date = $recur->dates();

# To find the Monday after Easter in 1997-1999

$recur = new Date::Manip::Recur;
$recur->parse("*1997-1999:0:0:0:0:0:0*EASTER,ND1");
@date = $recur->dates();
```

Functional

```
# To find the 2nd Tuesday of every month from Jan 1 1999 to Apr 30 1999
@date = ParseRecur("0:1*2:2:0:0:0","","Jan 1 1999","Apr 30 1999");

# To find the Monday after Easter in 1997-1999.
@date = ParseRecur("*1997-1999:0:0:0:0:0:0*EASTER,ND1");
```

The Date::Manip::Recur manual contains information about recurring events.

WORKING WITH DATES IN ANOTHER LANGUAGE

If you want to work with dates in a language other than English (but you are only working with a single language), do the following:

OO method

```
$date = new Date::Manip::Date;
$date->config("Language","French","DateFormat","non-US");
$date->parse("1er decembre 1990");
```

Functional

```
Date_Init("Language=French","DateFormat=non-US");
$date = ParseDate("1er decembre 1990");
```

The Date::Manip::Config manual has a list of all supported languages (in the section on the Language config variable). The meaning of the DateFormat config variable is also included.

WORKING WITH TWO DIFFERENT LANGUAGES

If you want to work with dates in two (or more) languages, it is STRONGLY recommended that you use the OO interface. The functional interface will be much slower since it has to re-initialize a lot of language-

specific stuff every time you switch back and forth between languages.

OO method

```
$date_eng = new Date::Manip::Date;
$date_eng->config("Language","English","DateFormat","US");

$date_fre = new Date::Manip::Date;
$date_fre->config("Language","French","DateFormat","non-US");
```

Use the `$date_eng` object to do English operations, the `$date_fre` object to do French operations.

Functional

If you are working with both French and English dates, you can call the following to switch between them:

```
Date_Init("Language=French","DateFormat=non-US");
Date_Init("Language=English","DateFormat=US");
```

This is NOT recommended. Use the OO method instead.

BUGS AND QUESTIONS

Please refer to the Date::Manip::Problems documentation for information on submitting bug reports or questions to the author.

SEE ALSO

Date::Manip − main module documentation

LICENSE

AUTHOR

Sullivan Beck (sbeck@cpan.org)