NAME

Date::Manip::Holidays − describes holidays and events

SYNOPSIS

This describes the Holidays and Events sections of the config file, and how they are used.

Holidays and events are specific days that are named. Holidays are used in business mode calculations, events are not. Events may be used for other calendaring operations.

HOLIDAYS

The holiday section of the config file is used to define holidays. Each line is of the form:

```
   STRING = HOLIDAY
```

HOLIDAY is the name of the holiday or it can be blank.

If HOLIDAY is blank, the holiday is unnamed, but still treated as a holiday. For example, in the US, the day after Thanksgiving is often a work holiday though it is not named.

HOLIDAY should be unique in most cases. The only exception is if the holiday definition is complex enough that it is impossible to describe it with one STRING. In this case, multiple lines may be given with different values of STRING but the same value for HOLIDAY, and in these cases, the first STRING that matches a given year will be used. This situation is described in more detail below.

NOTE: It is not allowed to have unnamed holidays that require multiple definitions, so a name will have to be assigned in that case.

STRING is a string which can be parsed to give a valid date. It can be any of the following forms:

A full date

Specific holidays can be set which occur only a single time.

```
   May 5, 2000                     = A one−time−only holiday
```

Any format parseable by `Date::Manip::Date::parse_date` can be used.

There is one caveat to using a full date. Date::Manip assumes that most holidays will appear once per year, so if you were to explicitly defined New Years (observed) as:

```
   2004−12−31                      = New Year's Day
```

then it would assume that it had found the occurrence of New Year’s for 2004 when in fact, this is the 2005 occurrence.

Full date specifications should only be used as a last resort, and probably only if you will explicitly specify all occurrence of the holiday.

A date without a year

Some holidays occur every year on the same day. These can be defined using the simple lines:

```
   Jan 1                           = New Year's Day
   Jul 4th                         = Independence Day
   fourth Thu in Nov               = Thanksgiving
```

These dates must be written in a form which can be parsed as a full date by simply adding the year to the end of the string. Please refer to the Date::Manip::Date documentation to see what forms will work. ISO 8601 dates will not work since the year comes first.

Any format parseable by `Date::Manip::Date::parse_date` which allows the year to be at the end can be used.

Recurrence

The dates can be specified using recurrences:

```
   1*0:0:0:0:0:0*EASTER            = Easter
   1*11:0:11:0:0:0*DWD             = Veteran's Day
   1*11:4:4:0:0:0                  = Thanksgiving
   1*11:4:4:0:0:0*FD1              = Day after Thanksgiving
```

In cases where you are interested in business type calculations, you'll want to define most holidays using recurrences, since they can define when a holiday is celebrated in the financial world. For example, Christmas might be defined as:

```
   Dec 25                = Christmas
```

but if it falls on a weekend, there won't be a business holiday associated with it. It could be defined using a recurrence:

```
   1*12:0:24:0:0:0*DWD  = Christmas
```

so that if Christmas falls on a weekend, a holiday will be taken on the Friday before or the Monday after the weekend.

You can use the fully specified format of a recurrence:

```
  1*2:0:1:0:0:0***Jan 1 1999*Dec 31 2002 = Feb 2 from 1999−2002
```

OTHER HOLIDAY CONSIDERATIONS

Recurrences which change years

It is now valid to have a recurrence defined for New Year's day which pushes the holiday to the previous year.

For example, the most useful definition of New Year's day is:

```
   1*1:0:1:0:0:0*DWD               = New Year's Day
```

which means to choose the closest working day to observe the holiday, even though this might mean that the holiday is observed on the previous year.

Order of definitions is preserved

The order of the definitions is preserved. In other words, when looking at the holidays for a year, previously defined holidays (in the order given in the config file) are correctly handled.

As an example, if you wanted to define both Christmas and Boxing days (Boxing is the day after Christmas, and is celebrated in some parts of the world), and you wanted to celebrate Christmas on a business day on or after Dec 25, and Boxing day as the following work day, you could do it in one of the following ways:

```
   1*12:0:25:0:0:0*NWD  = Christmas
   1*12:0:26:0:0:0*NWD  = Boxing
```

or

```
   1*12:0:25:0:0:0*NWD  = Christmas
   1*12:0:25:0:0:0*NWD  = Boxing
```

Holidays go into affect the minute they are parsed which is why the second example works (though for clarity, the first one is preferable). The first recurrence defined the first business day on or after Dec 25 as Christmas. The second one then defines the business day after that as Boxing day. Since the definitions are stored as a list (NOT a hash as they were in Date::Manip 5.xx), using the same recurrence twice does not cause a problem.

Multiple holidays

Having multiple holidays on a single day is allowed. As an example, you may want to look at New Years day as both the observed and actual holidays, so you might have:

```
   1*1:0:1:0:0:0*DWD                = New Year's Day (observed)
   Jan 1                            = New Year's Day
```

Most of the time, both will fall on the same day, but sometimes they may differ. In this example, it is important that the observed holiday be listed first. Otherwise, Jan 1 will be marked as a holiday and then the observed date will check Jan 1, but where it is not a business day, it will move to another day (due to the DWD modifier).

Likewise, the two holidays:

```
   3rd Sunday in June               = Father's Day
   Jun 17                           = Bunker Hill Day
```

sometimes fall on the same day. Using the`Date::Manip::Date::list_holidays` method (or the `Date_IsHoliday` function), you can get a list of all names that the date contains.

Complex holiday descriptions

Occasionally, you cannot describe a holiday using a single line. For example, the US Federal Reserve banks use a complex holiday description where:

```
   For holidays falling on Saturday, Federal Reserve Banks
   and Branches will be open the preceding Friday. For holidays
   falling on Sunday, all Federal Reserve Banks and Branches
   will be closed the following Monday.
```

Since Saturday is not a business day, the DWD modifier will not work. For these, you need a more complicated definition.

The following definitions both work:

```
   # Saturday
   1*1:0:1:0:0:0*NBD,BD1,IBD,FD1   = New Year's Day
   # Sunday (observed Monday)
   1*1:0:1:0:0:0*NBD,BD1,NBD,FD2   = New Year's Day
   # M−F
   1*1:0:1:0:0:0*IBD               = New Year's Day
```

and

```
   # Saturday
   1*1:0:1:0:0:0*IW6               = New Year's Day
   # Sunday (observed Monday)
   1*1:0:1:0:0:0*IW7,FD1           = New Year's Day
   # M−F
   1*1:0:1:0:0:0*IBD               = New Year's Day
```

EVENTS

The Events section of the config file is similar to the Holiday section. It is used to name certain days or times, but there are a few important differences:

Events can be assigned to any time and duration

All holidays are exactly 1 day long. They are assigned to a period of time from midnight to midnight.

Events can be based at any time of the day, and may be of any duration.

Events don't affect business mode calculations

Unlike holidays, events are completely ignored when doing business mode calculations.

Whereas holidays were added with business mode math in mind, events were added with calendar and scheduling applications in mind.

Every line in the events section is of the form:

```
EVENT = NAME
```

where NAME is the name of the event, and EVENT defines when it occurs and its duration. An EVENT can be defined in the following ways:

```
Date
YMD
YM
Recur

Date  ; Date
YMD   ; YMD
YM    ; YM
Date  ; Delta
Recur ; Delta
```

Date refers to a full date/time (and is any string that can be parsed by `Date::Manip::Date::parse`). YMD is any string which can be parsed by `Date::Manip::Date::parse_date`. YM is any string which can be parsed by the parse_date method to give a date in the current year. Recur is a partial or fully specified recurrence. Delta is any string that can be parsed to form a delta.

With the ‘‘Date’’ form, or the ‘‘Recur’’ form, the event starts at the time (or times) specified by the date or recurrence, and last 1 hour long. With the ‘‘YMD’’ and ‘‘YM’’ forms, the event occurs on the given day, and lasts all day.

With all of the two part forms (‘‘Date;Date’’, ‘‘YM;YM’’, etc.), the event starts at the first date and goes to the second date, or goes an amount of time specified by the delta.

The ‘‘YMD;YMD’’ and ‘‘YM;YM’’ forms means that the event lasts from the start of the first date to the end of the second. In the Date;Date form, the event goes from the first date to the second date inclusive. In other words, both dates are in the event. In the ‘‘Date;Delta’’ and ‘‘Recur;Delta’’ forms, the Delta tells the length of the event. Also, in the Date;Date form, the second date may NOT be expressed as a delta.

Currently, having an event longer than 1 year is NOT supported, but no checking is done for this.

KNOWN BUGS

None known.

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)