DATE/TIME TYPE IN LPL

Tony Hürlimann

Working Paper

2003-10-24T09:10:15

Date/Time Type in LPL

One of the Dates was : 14th of May 98, 8h15 AM in the Thames

Does killing time damage eternity? (*Johann Wolfgang Von Goethe)*

Everything's temporary if you give it enough time. (*John Newton)*

...but even the Emperor himself cannot buy another day. (*Albert Einstein)*

Keep a diary, and someday it'll keep you.

Introduction

Time plays an important role in various model-applications. Examples are found in the areas of control, scheduling and planning. The time scale in control is typically seconds and minutes, in scheduling it is hours or days, and in planning it is days, weeks and months or years. The ISO 8601 standard has defined an international Date/Time Format, which is used as the base of LPL's new DATE data type.

ISO 8601

International Standard ISO 8601 specifies numeric representations of date and time. This standard notation helps to avoid confusion in international communication caused by the many different national notations and increases the portability of computer user interfaces. In addition, these formats have several important advantages for computer usage compared to other traditional date and time notations. The time notation described here is already the de-facto standard in almost all countries and the date notation is becoming increasingly popular [Kuhn].

Date

The international standard date notation is

YYYY-MM-DD

where YYYY is the year in the usual Gregorian calendar, MM is the month of the year between 01 (January) and 12 (December), and DD is the day of the month between 01 and 31.

For example, the fourth day of February in the year 1995 is written in the standard notation as

1995-02-04

Advantages of the ISO 8601 standard date notation compared to other commonly used variants:

- easily readable and writeable by software (no 'JAN', 'FEB', ... table necessary)
- easily comparable and sortable with a trivial string comparison
- language independent
- can not be confused with other popular date notations
- consistency with the common 24h time notation system, where the larger units (hours) are also written in front of the smaller ones (minutes and seconds)
- strings containing a date followed by a time are also easily comparable and sortable (e.g. write "1995-02-04 22:45:00")
- the notation is short and has constant length, which makes both keyboard data entry and table layout easier
- identical to the Chinese date notation, so the largest cultural group (>25%) on this planet is already familiar with it :-)
- date notations with the order "year, month, day" are in addition already widely used e.g. in Japan, Korea, Hungary, Sweden, Finland, Denmark, and a few other countries and people in the U.S. are already used to at least the "month, day" order

- a 4-digit year representation avoids overflow problems after 2099-12-31

ISO 8601 is only specifying numeric notations and does not cover dates and times where words are used in the representation. It is not intended as a replacement for language-dependent worded date notations such as "24. Dezember 2001" (German) or "February 4, 1995" (US English). ISO 8601 should however be used to replace notations such as "2/4/95" and "9.30 p.m.".

Apart from the recommended primary standard notation **YYYY-MM-DD**, ISO 8601 also specifies a number of alternative formats for use in applications with special requirements. All of these alternatives can easily and automatically be distinguished from each other:

The hyphens can be omitted if compactness of the representation is more important than human readability, for example as in

19950204

For situations where information about the century is really not required, a 2-digit year representation is available:

95-02-04 or **950204**

If only the month or even only the year is of interest:

1995-02 or **1995**

In commercial and industrial applications (delivery times, production plans, etc.), especially in Europe, it is often required to refer to a week of a year. Week 01 of a year is per definition the first week that has the Thursday in this year, which is equivalent to the week that contains the fourth day of January. In other words, the first week of a new year is the week that has the majority of its days in the new year. Week 01 might also contain days from the previous year and the week before week 01 of a year is the last week (52 or 53) of the previous year even if it contains days from the new year. A week starts with Monday (day 1) and ends with Sunday (day 7). For example, the first week of the year 1997 lasts from 1996-12-30 to 1997-01-05 and can be written in standard notation as

1997-W01 or **1997W01**

The week notation can also be extended by a number indicating the day of the week. For example, the day 1996-12-31, which is the Tuesday (day 2) of the first week of 1997, can also be written as

1997-W01-2 or **1997W012**

for applications like industrial planning where many things like shift rotations are organized per week and knowing the week number and the day of the week is more handy than knowing the day of the month.

An abbreviated version of the year and week number like

95W05

is sometimes useful as a compact code printed on a product that indicates when it has been manufactured.

Both day and year are useful units of structuring time, because the position of the sun on the sky, which influences our lives, is described by them. However the 12 months of a year are of some obscure mystic origin and have no real purpose today except that people are used to having them (they do not even describe the current position of the moon). In some applications, a date notation is preferred that uses only the year and the day of the year between 001 and 365 (366 in leap years). The standard notation for this variant representing the day 1995-02-04 (that is day 035 of the year 1995) is

1995-035 or **1995035**

Leap years are years with an additional day YYYY-02-29, where the year number is a multiple of four with the following exception: If a year is a multiple of 100, then it is only a leap year if it is also a multiple of 400. For example, 1900 was not a leap year, but 2000 is one.

Time

The international standard notation for the time of day is

hh:mm:ss

where hh is the number of complete hours that have passed since midnight (00-24), mm is the number of complete minutes that have passed since the start of the hour (00-59), and ss is the number of complete seconds since the start of the minute (00-60). If the hour value is 24, then the minute and second values must be zero. [The value 60 for ss might sometimes be needed during an inserted leap second in an atomic time scale like Coordinated Universal Time (UTC). A single leap second 23:59:60 is inserted into the UTC time scale every few years as announced by the International Earth Rotation Service in Paris to keep UTC from wandering away more than 0.9 s from the less constant astronomical time scale UT1 that is defined by the actual rotation of the earth.]

An example time is

23:59:59

which represents the time one second before midnight.

As with the date notation, the separating colons can also be omitted as in

235959

and the precision can be reduced by omitting the seconds or both the seconds and minutes as in

23:59, **2359**, or **23**

It is also possible to add fractions of a second after a decimal dot or comma, for instance the time 5.8 ms before midnight can be written as

23:59:59.9942 or **235959.9942**

As every day both starts and ends with midnight, the two notations **00:00** and **24:00** are available to distinguish the two midnights that can be associated with one date. This means that the following two notations refer to exactly the same point in time:

1995-02-04 24:00 = **1995-02-05 00:00**

In case an unambiguous representation of time is required, 00:00 is usually the preferred notation for midnight and not 24:00. Digital clocks display 00:00 and not 24:00.

ISO 8601 does not specify, whether its notations specify a point in time or a time period. This means for example that ISO 8601 does not define whether 09:00 refers to the exact end of the ninth hour of the day or the period from 09:00 to 09:01 or anything else. The users of the standard must somehow agree on the exact interpretation of the time notation if this should be of any concern.

If a date and a time are displayed on the same line, then always write the date in front of the time. If a date and a time value are stored together in a single data field, then ISO 8601 suggests that they should be separated by a latin capital letter T, as in **19951231T235959**.

In the U.S., the widely respected *Chicago Manual of Style* now recommends using the international standard time notation in publications.

The German standard DIN 5008, which specifies typographical rules for German texts written on typewriters, was updated in 1996-05. The old German numeric date notations DD.MM.YYYY and DD.MM.YY have been replaced by the ISO date notations YYYY-MM-DD and YY-MM-DD. Similarly, the old German time notations hh.mm and hh.mm.ss have been replaced by the ISO notations hh:mm and hh:mm:ss. Those new notations are now also mentioned in the latest edition of the *Duden*. The German alphanumeric date notation continues to be for example "3. August 1994" or "3. Aug. 1994". The corresponding Austrian standard has already used the ISO 8601 date and time notations before.

ISO 8601 has been adopted as European Standard EN 28601 and is therefore now a valid standard in all EU countries and all conflicting national standards have been changed accordingly.

LPL's Date Type

LPL adopts the international Standard ISO 8601 of date and time formatting in its language specification with one exception: a date/time string must begin with the character '@'. So the character sequence @2003-10-07 is a legal date in LPL for 2003-10-07. It was necessary to adopt this convention, because otherwise the LPL parser interprets this as five tokens formed to an numerical expression which results in 1986 (2003 minus 10 minus 7).

An alternative would have been to modify the ISO 8601 slightly and replace the dashes by dots as in 2003.10.07. In this way any language parser using numerical tokens could recognize it as a date item. Because scanning a stream of digits then a dot and again stream of digits is in most computer languages interpreted as a real number. Adding still another dot with a stream of digit would be easy for such a parser to interpret the stream as a date. It is a pity that the ISO 8601 standard did not taken this language requirements into account. As is, we prefer to adopt the standard with the inconvenience of the intial character '@'.

Of course, this is only the representation of a date within the LPL source text. The internal representation is different. The representation of a date within a database als has nothing to do with this representation proposed here. Reading from and writing date/timeto a database follows the specification of the database system. A text field with strings like '@2002-03-01' will not be recognized by the database LPL reader as a date, although there exists a function in LPL that can translate a string like the previous one into a date type. The read dates from a database it is enough to defines the database field as a date/time field and the corresponding LPL paramanter as a DATE PARAMETER. (DATE is a new keyword in LPL like INTEGER or BINARY and indicates the type of an entity.)

The syntax in LPL source code is as follows :

```
Date = '@' yyyy   [ '-' MM [ '-' dd] ]
Time = hh [ ':' mm [ ':' ss ] ]
DateTime = Date ['T' Time ]
```

- yyyy, MM, dd, hh, mm, ss are all digits.
- yyyy can be in the range [1899..9999] and defines the year.
- MM is in the range of [01..12] and designates the month.
- dd defines the day within the month and is in the range [01..31]. However, 1999-04-31 will not be accepted as a legal date, neither will be 2001-02-29.
- hh defines the hour within a day and ranges in [00..23]
- mm defines the minutes and ranges in [00..59]
- ss defines the seconds and its range is [00..59].

Every date/time specification in an LPL code is interpreted as a time *point*. So :

@2003-10-07 means 2003-10-07T00:00:00

@2003-10	means	2003-10-01T00:00:00
@2003-W03-2	means	2003-01-14T00:00:00
@2003W04	means	2003-01-20T00:00:00

In LPL it is not a legal date to remove the dashes as suggested in the ISO standard for more compact writing.

The internal representation of a date/time is a double (64bit) it has the same format as Delphi's internal representation of the type *TDateTime*. This is extremely convenient. It allows to calculate in a limited ways using date like numbers. Everywhere where numbers are allowed in LPL also dates can be used and vice-versa.

The following operations are semantically useful:

Difference of two dates : just substract an date from another (the smaller from the larger). In LPL :

```
PARAMETER number_of_days = date1 - date2;
```

Adding or substracting a number from a date : In LPL :

```
DATE PARAMETER new_date = date1 - number_of_days;
DATE PARAMETER new_date = date1 + number_of_days;
```

Inversely every double (numerical value) can be interpreted as a date/time point supposing that the parameter is defined as a DATE type.

Dates are also allows in SETs. Defining an set as :

```
date set i := / @2003-10-01:@2003-10-10 /;
```

generates the date set

```
2003-10-01
2003-10-02
2003-10-03
2003-10-04
2003-10-05
2003-10-06
2003-10-07
2003-10-08
2003-10-09
```

Delphi's definition of the internal date/time fomat

Date and time values using a *TDateTime* type are internally defined as *double*. (In C++ also, the TDateTime class corresponds to the Delphi TDateTime type.)

The integral part of a Delphi TDateTime value is the number of days that have passed since 1899-12-30. The fractional part of the TDateTime value is fraction of a 24 hour day that has elapsed.

Following are some examples of TDateTime values and their corresponding dates and times:

0	1899-12-30T12:00
2.75	1900-01-01T18:00
-1.25	1899-12-29T06:00
35065	1996-01-01T12:00

To find the fractional number of days between two dates, simply subtract the two values, unless one of the *TDateTime* values is negative. Similarly, to increment a date and time value by a certain fractional number of days, add the fractional number to the date and time value if the *TDateTime* value is positive.

When working with negative *TDateTime* values, computations must handle time portion separately. The fractional part reflects the fraction of a 24-hour day without regard to the sign of the *TDateTime* value. For example, 1899-12-29T06:00 is –1.25, not –1 + 0.25, which would be –0.75. There are no *TDateTime* values between –1 and 0 [Delphi].

Bibliographie

Kuhn., http://www.cl.cam.ac.uk/~mgk25/iso-time.html.

Delphi., Version 7, see Datatype *TdateTime* in the Help System of Delphi.