Article
Erik Hemdal · Oct 31, 2016 8m read

Handling Date and Time Operations in Caché

Here are a few examples of conversions and operations you might need, along with links to documentation where you can learn more.

At the time I wrote this, Eastern Daylight Time was in effect for my Caché system.

How Caché keeps the time and date

Caché has a simple time format, with a longer range of recognized dates compared to some other technologies.

The current time is maintained in a special variable $HOROLOG ($H):

```
USER>WRITE $H

64146,54027

USER>
```

The first integer is the count of days since December 31, 1840. The second integer is the count of seconds since midnight of the current day.

You can also obtain the current time and date with $SYSTEM.SYS.Horolog().

How to establish a timestamp

$HOROLOG tracks time down to the second. $ZTIMESTAMP is similar in form to $HOROLOG, but it tracks fractions of a second in the time portion and it keeps Coordinated Universal Time (UTC), rather than local time. The precision of the fractional seconds depends on your platform.

Consequently, $ZTIMESTAMP provides a time stamp that is uniform across time zones. The timestamp you see at any point in time might have a date and time different from your current localtime. In this example, my localtime is Eastern Daylight Time, four hours behind UTC.

```
WRITE !,"$ZTIMESTAMP: "_$ZTIMESTAMP_" $HOROLOG: "_$HOROLOG

$ZTIMESTAMP: 64183,53760.475 $HOROLOG: 64183,39360
```

The difference (not counting the fractional seconds) is 14400 seconds. My $HOROLOG is thus four hours “ behind ” $ZTIMESTAMP.

How to convert from internal to display format

You can use $ZDATETIME. Converting the current date and time from internal format can be as simple as

```
WRITE !, "With default date and time format: ",$ZDATETIME($HOROLOG)

With default date and time format: 09/22/2016 10:56:00
```

This takes the default settings from the Caché locale and NLS settings.

The second and third (optional) arguments are for specifying the date format and time format.

```
WRITE !, "With dformat 5 and tformat 5: ", $ZDATETIME($HOROLOG,5,5)

With dformat 5 and tformat 5: Sep 22, 2016T10:56:00-04:00
```

Time format 7, for example, displays the time as Coordinated Universal Time as you see here.

```
WRITE !, "With dformat 5 and tformat 7: ", $ZDATETIME($HOROLOG,5,7)

With dformat 5 and tformat 7: Sep 22, 2016T14:56:00Z
```

Besides the date and time formats, there are many more optional arguments that give you control over the display. For example, you can

- Specify the lower and upper limits of valid dates if they are also allowed by the current locale.
- Control whether years are displayed with two or four digits.
- Control how errors are displayed.

How to convert between a display format and Caché internal format

Use $ZDATETIMEH to go from display format to internal format, like $HOROLOG. The H at the end is a reminder that you ’ ll end up with $HOROLOG format. Many different input formats can be used.

```
  SET display = "09/19/2016 05:05 PM"
  WRITE !, display_" is "_$ZDATETIMEH(display)_" in internal format"
  WRITE !, "Suffixes AM, PM, NOON, MIDNIGHT can be used"
  SET startdate = "12/31/1840 12:00 MIDNIGHT"
  WRITE !, startdate_" is "_$ZDATETIMEH(startdate)_" in internal format"


09/19/2016 05:05 PM is 64180,61500 in internal format

Suffixes AM, PM, NOON, MIDNIGHT can be used

12/31/1840 12:00 MIDNIGHT is 0,0 in internal format
```

How to convert from UTC to and from local time in internal format

You can use $ZDATETIME and $ZDATETIMEH with a special date-format specifier (-3) for the second argument.

The preferred way to convert UTC time to local time in Caché internal format is to use the $ZDATETIMEH(datetime, -3) function. Here, the first argument contains the UTC time in internal format.

```
  SET utc1 = $ZTIMESTAMP
```

```
  SET loctime1 = $ZDATETIMEH(utc1, -3)
  WRITE !, "$ZTIMESTAMP returns a UTC time in internal format: ", utc1
  WRITE !, "$ZDATETIMEH( ts,-3) converts UTC to local time: ", loctime1
  WRITE !, "$ZDATETIME converts this to display formats: ", $ZDATETIME(utc1)
  WRITE !, "which is "_$ZDATETIME(loctime1)_" in local time"

$ZTIMESTAMP returns a UTC time in internal format: 64183,53760.475

$ZDATETIMEH( ts,-3) converts UTC to local time: 64183,39360.475

$ZDATETIME converts this to display formats: 09/22/2016 14:56:00

which is 09/22/2016 10:56:00 in local time
```

If you need to go from local time to UTC, again in internal format, use $ZDATETIME(datetime, -3). Here, datetime contains the time reflecting your local timezone in internal format.

```
  SET loctime2 = $HOROLOG
  SET utc2 = $ZDATETIME(loctime2, -3)
  WRITE !, "$HOROLOG returns a local time in internal format: ", loctime2
  WRITE !, "$ZDATETIME(ts, -3) converts this to UTC: ", utc2
  WRITE !, "$ZDATETIME converts this to display formats:"
  WRITE !, "Local: ", $ZDATETIME(loctime2)
  WRITE !, "UTC: ", $ZDATETIME(utc2)

$HOROLOG returns a local time in internal format: 64183,39360

$ZDATETIME(ts, -3) converts this to UTC: 64183,53760

$ZDATETIME converts this to display formats:

Local: 09/22/2016 10:56:00

UTC: 09/22/2016 14:56:00
```

Keep these points in mind when performing local and UTC time conversions:

- Conversions between local time and UTC must use the timezone rules in effect for the specified date and location. Caché depends on the operating system to track those changes over time. If the operating system doesn ’ t do this properly, the conversions will not be correct.
- Conversions of future dates and times use the current rules maintained by the operating system. However, the rules for future years may change.

How to determine the system time zone

You can get the current timezone offset by examining the value of $ZTIMEZONE or %SYSTEM.SYS.TimeZone(). The default is set by the operating system.

```
 WRITE !, "$ZTIMEZONE is set to "_$ZTIMEZONE
 WRITE !, "%SYSTEM.SYS.TimeZone() returns "_$System.SYS.TimeZone()

$ZTIMEZONE is set to 300
```

```
%SYSTEM.SYS.TimeZone() returns 300
```

You should not change the value of $ZTIMEZONE. If you do, you will affect the results of IsDST(), $ZDATETIME, and $ZDATETIMEH, among many other effects. Time for the process will not change the hour properly for Daylight Saving Time. Changing $ZTIMEZONE is not a consistent way to change the timezone that Caché uses.

As of version 2016.1, Caché provides the $System.Process.TimeZone() method which allows you to set and retrieve the timezone for a specific process using the TZ environment variable. It returns -1 if TZ is not set.

```
WRITE !,$System.Process.TimeZone()
WRITE !, "Current Time: "_$ZDT($H)
WRITE !, "Set Central Time"
DO $System.Process.TimeZone("CST6CDT")
WRITE !, "New current time: "_$ZDT($H)
WRITE !, "Current Time Zone: "_$System.Process.TimeZone()


-1

Current Time: 10/03/2016 15:46:04

Set Central Time

New current time: 10/03/2016 14:46:04

Current Time Zone: CST6CDT
```

How to determine if Daylight Saving Time is in effect

Use $SYSTEM.Util.IsDST(). Here, too, Caché relies on the operating system to apply the correct rules for determining if Daylight Saving Time is in effect.

```
  SET dst = $System.Util.IsDST()
  IF (dst = 1) {WRITE !, "DST is in effect"}
  ELSEIF (dst = 0) { WRITE !, "DST is not in effect" }
  ELSE { WRITE !, "DST cannot be determined" }
```

How to perform date arithmetic

Since the Caché internal format maintains a count of days and a count of seconds within each day, you can do date arithmetic in a straightforward way. The $PIECE function lets you take apart the date and time parts of the internal format.

Here is a short routine which uses $ZDATE and $ZDATEH to determine the last day of last year so you can count today ' s day-of-the-year. This uses methods in class %SYS.NLS to set the date format to what we want, obtain the date separator, and set things back to the default.

```
DATECALC ; Example of date arithmetic.
  W !, "Extracting date and time from $H using $PIECE"
  W !, "-------------------------------------------"
  set curtime = $H
```

```
  set today = $PIECE(curtime,",",1)
  set now = $PIECE(curtime,",",2)
  W !, "Curtime: "_curtime_" Today: "_today_" Now: "_now

  W !, "Counting the days of the year"
  W !, "----------------------------"
  ; set to US format
  SET rtn = ##class(%SYS.NLS.Format).SetFormatItem("DateFormat",1)
  set sep = ##class(%SYS.NLS.Format).GetFormatItem("DateSeparator")
  SET lastyear = ($PIECE($ZDATE($H),sep,3) - 1)
  SET start = $ZDATEH("12/31/"_lastyear)
  W !, "Today is day "_(today - start)_" of the year"
  ; put back the original date format
  SET rtn=##class(%SYS.NLS.Format).SetFormatItem("DateFormat",rtn)
```

How to get and set other NLS settings

Use the class %SYS.NLS.Format for settings such as date format, maximum and minimum dates and other settings. Initial settings come from the current locale and changes you make with this class affect the current process only.

Time and Date in SQL

Caché provides a variety of SQL functions for working with dates and times. These are also available in ObjectScript via class $System.SQL.

TODATE: Converts a date in CHAR or VARCHAR2 format to a date. This is available in ObjectScript using $System.SQL.TODATE(“ string ” , ” format ”)

DAYOFYEAR: Returns the day-of-the-year for a given year expression, which can be in several formats, such as a date integer from $HOROLOG.

DAYNAME: returns the name of the day that corresponds to a specified date.

```
W $ZDT($H)

10/12/2016 11:39:19


w $System.SQL.TODATE("2016-10-12","YYYY-MM-DD")

64203


W $System.SQL.DAYOFYEAR(+$H)

286


W $System.SQL.DAYNAME(+$H)

Wednesday
```

There is much more information in Cache ' documentation about how to use these functions (and many more). See the references in the next section.

References

Links to InterSystems online documentation for the items discussed here.

$HOROLOG: http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOSvhorolog

$PIECE: http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOSfpiece

SQL Functions reference:
http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RSQLFUNCTIONS

%SYS.NLS.Format:
http://docs.intersystems.com/latest/csp/documatic/%25CSP.Documatic.cls?PAGE=CLASS&LIBRARY=%25SYS&CLASSNAME=%25SYS.NLS.Format

%SYSTEM.Process.TimeZone():
http://docs.intersystems.com/latest/csp/documatic/%25CSP.Documatic.cls?PAGE=CLASS&LIBRARY=%25SYS&CLASSNAME=%25SYSTEM.Process#METHODTimeZone

%SYSTEM.SQL:
http://docs.intersystems.com/latest/csp/documatic/%25CSP.Documatic.cls?PAGE=CLASS&LIBRARY=%25SYS&CLASSNAME=%25SYSTEM.SQL

%SYSTEM.SYS.Horolog:
http://docs.intersystems.com/latest/csp/documatic/%25CSP.Documatic.cls?PAGE=CLASS&LIBRARY=%25SYS&CLASSNAME=%25SYSTEM.SYS

%SYSTEM.Util.IsDST():
http://docs.intersystems.com/latest/csp/documatic/%25CSP.Documatic.cls?PAGE=CLASS&LIBRARY=%25SYS&CLASSNAME=%25SYSTEM.Util#IsDST

$ZDATE: http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOSfzdate

$ZDATEH: http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOSfzdateh

$ZDATETIME: http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOSfzdatetime

$ZDATETIMEH: http://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=RCOSfzdatetimeh

#Best Practices #ObjectScript #System Administration #Caché