

srfi-19 egg

Provides SRFI-19.
Extension for Chicken Scheme
Version 1.6

Kon Lovett

Table of Contents

1	About this egg	1
1.1	Version history	1
1.2	Requirements	1
1.3	Usage	1
2	Documentation	2
2.1	Extensions to SRFI-19	2
2.1.1	Notes	2
2.1.2	Procedures	2
3	Issues	5
4	License	6
	Index	7

1 About this egg

1.1 Version history

- 1.6 Bug fix for inexact seconds in time->date [thanks to Peter Bex]
- 1.5 Bug fix for compiled use
- 1.4 Exports
- 1.3 Bug fix
- 1.2 Slightly smaller and faster
- 1.1 Some srfi-18 conflict reduction
- 1.0 Initial release

1.2 Requirements

This egg requires the following extensions:

`locale`, `srfi-29`, `vector-lib`, `numbers`

1.3 Usage

Load this egg like so:

```
(require-extension srfi-19)
```

2 Documentation

srfi-19 is a CHICKEN port of SRFI-19.

This package largely follows the specification of SRFI-19. For more information see the documentation for [SRFI-19](#).

2.1 Extensions to SRFI-19

2.1.1 Notes

The `nanosecond` time object element is an integer between 0 and 999,999,999 inclusive. (Not an extension, the srfi document is just wrong.)

The procedure `srfi-19:current-time` is a synonym for the `current-time` procedure.

The procedure `srfi-19:time?` is a synonym for the `time?` procedure.

The `string->date` procedure allows the template-name argument to be optional. When missing the locale's date-time-format string is used. The supplied locale bundle's strings are invertible.

The `make-date` procedure allows an optional timezone abbreviation name as the last argument.

Where the srfi document states a `tz-offset` argument a timezone-locale structure is allowed, in addition to an offset value. A timezone-locale structure is a pair where the car is a boolean stating whether daylight saving time (summer time) is in effect, and the cdr is a timezone-components object.

Be careful using the procedures that return some form of 'julian-day.' These are implemented using the full numeric tower and *will* return rational numbers. Performing arithmetic with such a result will require the "numbers" egg. See the file "srfi-19-test.scm" in this egg for an example.

2.1.2 Procedures

`read-leap-second-table` [procedure]

(`read-leap-second-table` [FILENAME])

Sets the leap second table from the specified FILENAME. When missing the (repository-path) "tai-utc.dat" filename is used. (Missing from the srfi document.)

`leap-year?` [procedure]

(`leap-year?` DATE)

Is the specified DATE year a leap year? (Not an extension, just missing from the srfi document.)

`time->srfi-18-time` [procedure]

(`time->srfi-18-time` TIME)

Converts a srfi-19 time object to a srfi-18 time object. The conversion is really only meaningful for time-duration, but any time-type is accepted.

- srfi-18-time->time** [procedure]
 (srfi-18-time->time TIME)
 Converts a srfi-18 time object into a srfi-19 time-duration object.
- seconds->time/type** [procedure]
 (seconds->time/type SECONDS [TIME-TYPE])
 Converts a SECONDS value, may be fractional, into a TIME-TYPE time object. The default time-type is `time-duration`.
- seconds->date/type** [procedure]
 (seconds->date/type SECONDS [DATE-TYPE])
 Converts a SECONDS value, may be fractional, into a date object. The DATE-TYPE is `#t` for the local timezone or `#f` for the utc timezone. The default date-type is `#f`.
- format-date** [procedure]
 (format-date DESTINATION DATE-FORMAT-STRING [DATE])
 Displays a text form of the DATE on the DESTINATION using the DATE-FORMAT-STRING. When the destination is `#t` the current-output-port is used, and the date object must be specified. When the destination is a string the date-format-string value must be a date object, the destination value is used as the date-format-string, and the result is returned as a string. When the destination is a port it must be an output-port, and the date object must be specified. When the destination is a number the current-error-port is the destination, and the date object must be specified. When the destination is `#f` the result is returned as a string, and the date object must be specified.
- scan-date** [procedure]
 (scan-date SOURCE TEMPLATE-STRING)
 Reads a text form of a date from the SOURCE, following the TEMPLATE-STRING, and returns a date object. When the source is `#t` the current-input-port is used. When the source is a port it must be an input-port. When the source is string it should be a date text form.
- copy-date** [procedure]
 (copy-date DATE)
 Returns an exact copy of the specified DATE object.
- date-zone-name** [procedure]
 (date-zone-name DATE)
 Returns the timezone abbreviation of the specified DATE object. The result is either a string or `#f`.
- time->nanoseconds** [procedure]
 (time->nanoseconds TIME)
 Returns the TIME object value as a nanoseconds value.
- nanoseconds->time** [procedure]
 (nanoseconds->time NANOSECONDS [TIME-TYPE])
 Returns the NANOSECONDS value as a time TIME-TYPE object. The default time-type is `time-duration`.

`nanoseconds->seconds` [procedure]

`(nanoseconds->seconds NANOSECONDS)`

Returns the `NANOSECONDS` value as an inexact seconds value.

`milliseconds->time` [procedure]

`(milliseconds->time MILLISECONDS [TIME-TYPE])`

Returns the `MILLISECONDS` value as a time `TIME-TYPE` object. The default time-type is `time-duration`.

`time->date` [procedure]

`(time->date TIME)`

Returns the `TIME` object value as a date. A shorthand for the `(time-->date ...)` procedures.

3 Issues

The SRFI-18 `current-time` and `time?` procedures conflict with this srfi's procedures.

The SRFI-18 time object is not accepted except by the conversion procedures.

On a non-GNU (`build-platform`) there is no default local timezone abbreviation. I hope to rectify this in the future.

The expression `(time=? (seconds->time/type (nanoseconds->seconds (time->nanoseconds a-time-duration))) a-time-duration)` might be `#f`, due to the use of inexact arithmetic.

While it is possible to coerce 'julian-day' results to an inexact number the "numbers" egg bindings will be active/loaded anyway. This can be a problem with a file compiled using 'generic-arithmetic'. I suggest an intermediate file that wraps any 'julian-day' calls and coerces to an inexact number. Use the wrapped 'julian-day' call in your 'generic-arithmetic' source.

4 License

Copyright (c) 2005, Kon Lovett. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the Software), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED ASIS, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Index

C

copy-date 3

D

date-zone-name 3

F

format-date 3

L

leap-year? 2

M

milliseconds->time 4

N

nanoseconds->seconds 4

nanoseconds->time 3

R

read-leap-second-table 2

S

scan-date 3

seconds->date/type 3

seconds->time/type 3

srfi-18-time->time 3

T

time->date 4

time->nanoseconds 3

time->srfi-18-time 2