

guile-r7rs

Introduction

guile-r7rs is the collection of libraries part of R7RS bundled for GNU Guile 2.2 or later.

Table of Content

R7RS small

- `(scheme base)` TODO: TEST + DOC
- `(scheme case-lambda)` TODO: CODE + TEST + DOC
- `(scheme char)` TODO: CODE + TEST + DOC
- `(scheme complex)` TODO: CODE + TEST + DOC
- `(scheme cxr)` TODO: CODE + TEST + DOC
- `(scheme eval)` TODO: CODE + TEST + DOC
- `(scheme file)` TODO: CODE + TEST + DOC
- `(scheme inexact)` TODO: CODE + TEST + DOC
- `(scheme lazy)` TODO: CODE + TEST + DOC
- `(scheme load)` TODO: CODE + TEST + DOC
- `(scheme process-context)` TODO: CODE + TEST + DOC
- `(scheme r5rs)` TODO: CODE + TEST + DOC
- `(scheme read)` TODO: CODE + TEST + DOC
- `(scheme repl)` TODO: CODE + TEST + DOC
- `(scheme time)` TODO: CODE + TEST + DOC
- `(scheme write)` TODO: CODE + TEST + DOC

R7RS Red Edition

- `(scheme box)` aka. SRFI 111, TODO: CODE + TEST + DOC
- `(scheme charset)` aka. SRFI 14, TODO: CODE + TEST + DOC
- `(scheme comparator)` aka. SRFI 128, TODO: CODE + TEST + DOC
- `(scheme ephemerons)` aka. SRFI 124, TODO: CODE + TEST + DOC
- `(scheme generator)` aka. SRFI 121, TODO: CODE + TEST + DOC
- `(scheme hash-table)` aka. SRFI 125, TODO: CODE + TEST + DOC
- `(scheme ideque)` aka. SRFI 134, TODO: CODE + TEST + DOC
- `(scheme ilist)` aka. SRFI 116, TODO: CODE + TEST + DOC
- `(scheme list)` aka. SRFI 1, TODO: CODE + TEST + DOC
- `(scheme list-queue)` aka. SRFI 117, TODO: CODE + TEST + DOC
- `(scheme lseq)` aka. SRFI 127, TODO: CODE + TEST + DOC
- `(scheme rlist)` aka SRFI 101, TODO: CODE + TEST + DOC
- `(scheme set)` aka. SRFI 113, TODO: CODE + TEST + DOC
- `(scheme sort)` aka. SRFI 132, TODO: CODE + TEST + DOC
- `(scheme stream)` aka. SRFI 41, TODO: CODE + TEST + DOC

- (**scheme text**) aka. SRFI 135, TODO: CODE + TEST + DOC
- (**scheme vector**) aka. SRFI 133, TODO: CODE + TEST + DOC

(scheme base)

-

TODO (missing r7rs?)

...

TODO

=>

TODO

else

TODO

*

TODO

(+ **number** ...)

Addition procedure.

-

TODO

/

TODO

<

TODO

<=

TODO

=

TODO

>

TODO

>=

TODO

abs

TODO

and

TODO

append

TODO

apply

TODO

assoc

TODO

assq

TODO

assv

TODO

begin

TODO

binary-port?

TODO

boolean=?

TODO

boolean?

TODO

bytevector

TODO

bytevector-append

TODO

bytevector-copy

TODO

bytevector-copy!

TODO

`bytevector-length`

TODO

`bytevector-u8-ref`

TODO

`bytevector-u8-set!`

TODO

`bytevector?`

TODO

`caar`

TODO

`cadr`

TODO

`call-with-current-continuation`

TODO

`call-with-port`

TODO

`call-with-values`

TODO

`call/cc`

TODO

car

TODO

case

TODO

cdar

TODO

cddr

TODO

cdr

TODO

ceiling

TODO

char->integer

TODO

char-ready?

TODO

char<=?

TODO

char<?

TODO

`char=?`

TODO

`char>=?`

TODO

`char>?`

TODO

`char?`

TODO

`close-input-port`

TODO

`close-output-port`

TODO

`close-port`

TODO

`complex?`

TODO

`cond`

TODO

`cond-expand`

TODO

`cons`

TODO

`current-error-port`

TODO

`current-input-port`

TODO

`current-output-port`

TODO

`define`

TODO

`define-record-type`

TODO

`define-syntax`

TODO

`define-values`

TODO

`denominator`

TODO

`do`

TODO

`dynamic-wind`

TODO

`eof-object`

TODO

`eof-object?`

TODO

`eq?`

TODO

`equal?`

TODO

`eqv?`

TODO

`(error [who] message . irritants)`

Raise an error.

`error-object-irritants`

TODO

`error-object-message`

TODO

`error-object?`

TODO

`even?`

TODO

`exact`

TODO

`exact-integer-sqrt`

TODO

`exact-integer?`

TODO

`exact?`

TODO

`expt`

TODO

`features`

TODO

`file-error?`

TODO

`floor`

TODO

`floor-quotient`

TODO

`floor-remainder`

TODO

`floor/`

TODO

`flush-output-port`

TODO

`for-each`

TODO

`gcd`

TODO

`get-output-bytevector`

TODO

`get-output-string`

TODO

`guard`

TODO

`if`

TODO

`include`

TODO

`include-ci`

TODO

`inexact`

TODO

`inexact?`

TODO

`input-port-open?`

TODO

`input-port?`

TODO

`integer->char`

TODO

`integer?`

TODO

`lambda`

TODO

`lcm`

TODO

`length`

TODO

```
let
TODO

let*
TODO

let*-values
TODO

let-syntax
TODO

let-values
TODO

letrec
TODO

letrec*
TODO

letrec-syntax
TODO

list
TODO

list->string
TODO
```

`list->vector`

TODO

`list-copy`

TODO

`list-ref`

TODO

`list-set!`

TODO

`list-tail`

TODO

`list?`

TODO

`make-bytevector`

TODO

`make-list`

TODO

`make-parameter`

TODO

`make-string`

TODO

make-vector

TODO

map

TODO

max

TODO

member

TODO

memq

TODO

memv

TODO

min

TODO

modulo

TODO

negative?

TODO

newline

TODO

```
not

TODO

null?

TODO

number->string

TODO

number?

TODO

numerator

TODO

odd?

TODO

open-input-bytevector

TODO

open-input-string

TODO

open-output-bytevector

TODO

open-output-string

TODO
```

or

TODO

`output-port-open?`

TODO

`output-port?`

TODO

`pair?`

TODO

`parameterize`

TODO

`peek-char`

TODO

`peek-u8`

TODO

`port?`

TODO

`positive?`

TODO

`procedure?`

TODO

`quasiquote`

TODO

`quote`

TODO

`quotient`

TODO

`raise`

TODO

`raise-continuable`

TODO

`rational?`

TODO

`rationalize`

TODO

`read-bytevector`

TODO

`read-bytevector!`

TODO

`read-char`

TODO

`read-error?`

TODO

`read-line`

TODO

`read-string`

TODO

`read-u8`

TODO

`real?`

TODO

`remainder`

TODO

`reverse`

TODO

`round`

TODO

`set!`

TODO

`set-car!`

TODO

set-cdr!

TODO

square

TODO

string

TODO

string->list

TODO

string->number

TODO

string->symbol

TODO

string->utf8

TODO

string->vector

TODO

string-append

TODO

string-copy

TODO

`string-copy!`

TODO

`string-fill!`

TODO

`string-for-each`

TODO

`string-length`

TODO

`string-map`

TODO

`string-ref`

TODO

`string-set!`

TODO

`string<=?`

TODO

`string<?`

TODO

`string=?`

TODO

`string>=?`

TODO

`string>?`

TODO

`string?`

TODO

`substring`

TODO

`symbol->string`

TODO

`symbol=?`

TODO

`symbol?`

TODO

`syntax-error`

TODO

`syntax-rules`

TODO

`textual-port?`

TODO

`truncate`

TODO

`truncate-quotient`

TODO

`truncate-remainder`

TODO

`truncate/`

TODO

`u8-ready?`

TODO

`unless`

TODO

`unquote`

TODO

`unquote-splicing`

TODO

`utf8->string`

TODO

`values`

TODO

`vector`

TODO

`vector->list`

TODO

`vector->string`

TODO

`vector-append`

TODO

`vector-copy`

TODO

`vector-copy!`

TODO

`vector-fill!`

TODO

`vector-for-each`

TODO

`vector-length`

TODO

`vector-map`

TODO

`vector-ref`

TODO

`vector-set!`

TODO

`vector?`

TODO

`when`

TODO

`with-exception-handler`

TODO

`write-bytevector`

TODO

`write-char`

TODO

`write-string`

TODO

`write-u8`

TODO

`zero?`

TODO