

srfi-27 egg

Sources of Random Bits
Extension for Chicken Scheme
Version 0.3

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	Signatures	2
2.2	Structures	3
2.3	Parameters	4
2.4	Distributions	4
2.5	Internals	6
3	Issues	10
4	Examples	11
5	License	12
	Index	13

1 About this egg

1.1 Version history

- 0.3 MWC pseudo-randomize, better distribution API
- 0.2 More
- 0.1 Initial release

1.2 Requirements

This egg requires the following extensions:

`numbers`, `structures`, `miscmacros`, `misc-extn`

1.3 Usage

Load this egg like so:

```
(require-extension srfi-27)
```

2 Documentation

The srfi-27 egg is a CHICKEN port of the SRFI-27 reference implementation.

This package follows the specification of SRFI-27. For more information see the documentation for [SRFI-27](#).

Unlike the SRFI-27 specification a random-structure is not "open" by default. To use the builtin random structure do the following: `(require-extension structures srfi-27) (open MRG32k3a)`.

All random number generators are randomized using the current entropy source.

2.1 Signatures

```
entropy-source-signature [signature]
  (entropy-source-signature make-entropy-source entropy-source?
    entropy-kind
    entropic-u8vector entropic-f64vector entropic-u8
    entropic-f64)

make-entropy-source
  (-> %entropy-source)

entropy-source?
  (-> any boolean)

entropy-kind
  (-> (or string symbol))

entropic-u8vector
  (-> fixnum #!optional u8vector u8vector)

entropic-f64vector
  (-> fixnum #!optional f64vector f64vector)

entropic-u8
  (-> fixnum)

entropic-f64
  (-> flonum)

random-source-signature [signature]
  (random-source-signature random-integer random-real
    default-random-source make-random-source random-source?
    random-source-kind random-source-log2-period
    random-source-maximum-range random-source-maximum-modulus
    random-source-state-ref random-source-state-set!
    random-source-randomize! random-source-pseudo-randomize!
    random-source-make-integers random-source-make-reals)

random-integer
  (-> (integer exact positive) (integer exact positive (<= 0 ?) (< ? ?1)))
```

```

random-real
    (-> (real inexact (< 0 ? 1)))

default-random-source
    %random-source

make-random-source
    (-> %random-source)

random-source?
    (-> any boolean)

random-source-kind
    (-> %random-source (or string symbol))

random-source-log2-period
    (-> %random-source fixnum)

random-source-maximum-modulus
    (-> %random-source integer)

random-source-maximum-range
    (-> %random-source integer)

random-source-state-ref
    (-> %random-source random-state)

random-source-state-set!
    (-> %random-source random-state void)

random-source-randomize!
    (-> %random-source void)

random-source-pseudo-randomize!
    (-> %random-source integer integer void)

random-source-make-integers
    (-> %random-source (-> (integer exact positive) (integer exact positive)
    (<= 0 ?) (< ? ?1))))

random-source-make-reals
    (-> %random-source #!optional (real inexact (< 0 ? 1)) (-> (real inexact
    (< 0 ? 1))))

```

2.2 Structures

```

current-seconds-entropy                                     [structure]
    (current-seconds-entropy (entropy-source-signature))
    Entropy from (current-seconds).

dev-random-entropy                                         [structure]
    (dev-random-entropy (entropy-source-signature))
    Entropy from /dev/random.

```

dev-urandom-entropy	[structure]
(dev-urandom-entropy (entropy-source-signature))	
Entropy from /dev/urandom.	
MRG32k3a	[structure]
(MRG32k3a (random-source-signature))	
Pierre L'Ecuyer's Multiple Recursive Generator 32k3a.	
MWC	[structure]
(MWC (random-source-signature))	
George Marsaglia's Multiply With Carry generator.	
random-source-structures	[procedure]
(random-source-structures)	
Returns a list of the known random-source-structures.	
entropy-source-structures	[procedure]
(entropy-source-structures)	
Returns a list of the known entropy-source-structures.	

2.3 Parameters

current-entropy-source-structure	[parameter]
(current-entropy-source-structure [ENTROPY-SOURCE-STRUCTURE])	
Returns or sets the default entropy source structure.	
current-random-source-structure	[parameter]
(current-random-source-structure [RANDOM-SOURCE-STRUCTURE])	
Returns or sets the default random source structure.	

2.4 Distributions

All distributions return 2 values. The first is a procedure of arity 0, the random distribution generator. The second is a procedure of arity 0, returning the configuration arguments as multiple-values, in the order they appear in the 'make-*' argument list. The 'report' procedure can be safely ignored.

make-uniform-random-integers	[procedure]
(make-uniform-random-integers [HIGH] [LOW] [UNIT] [SOURCE])	
Returns an integer random number generator.	
SOURCE	A %random-source or a random-source-structure. Default is the 'current-random-source-structure'.
HIGH	An integer or boolean. The largest desired random integer. Default is one less than maximum range for the source. When boolean the default is used.
LOW	An integer. The smallest desired random integer. Default is 0.
UNIT	An integer. The spacing between desired random integers. Default is 1.

make-uniform-random-reals [procedure]

(make-uniform-random-reals [UNIT] [SOURCE])

Returns a real random number generator.

SOURCE A %random-source or a random-source-structure. Default is the 'current-random-source-structure'.

UNIT A number in (0 1) or boolean. The spacing, default is a pretty small number. When boolean the default is used.

make-random-exponentials [procedure]

(make-random-exponentials [MEAN 1] [RAND])

Exponentially distributed random inexact reals.

make-random-normals [procedure]

(make-random-normals [MEAN 0] [STDDEV 1] [RAND])

Normal distributed random inexact reals.

make-random-triangles [procedure]

(make-random-triangles [SMALLEST 0] [PROBABLE 0.5] [LARGEST 1] [RAND])

Triangular distributed random inexact reals.

make-random-poissons [procedure]

(make-random-poissons [MEAN 1] [RAND])

Poisson distributed random inexact reals.

make-random-bernoullis [procedure]

(make-random-bernoullis [P 0.5] [RAND])

Bernoulli distributed random booleans.

make-random-binomials [procedure]

(make-random-binomials [T 1] [P 0.5] [RAND])

Binomial distributed random integers.

make-random-geometrics [procedure]

(make-random-geometrics [P 0.5] [RAND])

Geometric distributed random inexact reals.

make-random-lognormals [procedure]

(make-random-lognormals [MEAN 1] [STDDEV 0] [RAND])

Log normal distributed random inexact reals.

make-random-cauchys [procedure]

(make-random-cauchys [MEADIAN 0] [STDDEV 1] [RAND])

Cauchy distributed random inexact reals.

make-random-gammas [procedure]

(make-random-gammas [ALPHA 1] [THETA 1] [RAND])

Gamma distributed random inexact reals.

random-normal-vector! [procedure]
 (random-normal-vector! VECTOR [MEAN 0] [STDDEV 1] [RAND])

Fills the VECTOR with normal distributed real numbers. When just a random generator is specified it is assumed to be a source of random normals!

VECTOR A vector or 64vector.

RAND A random reals source. Default is 'make-uniform-random-reals'.

MEAN A number.

STDDEV A number.

random-hollow-sphere! [procedure]
 (random-hollow-sphere! VECTOR [MEAN 0] [STDDEV 1] [RAND])

Fills the VECTOR with inexact reals the sum of whose squares is equal to 1.0. When just a random generator is specified it is assumed to be a source of random normals!

VECTOR A vector or 64vector.

RAND A random reals source. Default is 'make-uniform-random-reals'.

MEAN A number.

STDDEV A number.

random-solid-sphere! [procedure]
 (random-solid-sphere! VECTOR [MEAN 0] [STDDEV 1] [RAND])

Fills the VECTOR with inexact reals the sum of whose squares is less than 1.0. When just a random generator is specified it is assumed to be a source of random normals!

VECTOR A vector or 64vector.

RAND A random reals source. Default is 'make-uniform-random-reals'.

MEAN A number.

STDDEV A number.

2.5 Internals

These are only of interest if implementing an entropy or random source.

%entropy-source [record]

%make-entropy-source [procedure]
 (%make-entropy-source kind u8 f64 u8vector f64vector)

kind Description of the entropy source. symbol or string

u8 Procedure to generate an entropic unsigned 8 bit integer. (-> number)

f64 Procedure to generate an entropic 64 bit floating-point. (-> number)


```

u8vector  Procedure to generate a vector of entropic unsigned 8 bit integer. (-> fixnum #!optional (u8vector (<= (u8vector-length ?) ?1))
          u8vector)

f64vector
          Procedure to generate a vector of entropic 64 bit floating-point.
          (-> fixnum #!optional (f64vector (<= (f64vector-length ?) ?1))
          f64vector)

%entropy-source?                                [procedure]
          (%entropy-source? OBJECT)
          Is OBJECT an %entropy-source?

%entropy-source-u8                              [procedure]
          (%entropy-source-u8 ENTROPY-SOURCE)
          Returns the u8 procedure.

%entropy-source-f64                            [procedure]
          (%entropy-source-f64 ENTROPY-SOURCE)
          Returns the f64 procedure.

%entropy-source-u8vector                       [procedure]
          (%entropy-source-u8vector ENTROPY-SOURCE)
          Returns the u8vector procedure.

%entropy-source-f64vector                     [procedure]
          (%entropy-source-f64vector ENTROPY-SOURCE)
          Returns the f64vector procedure.

%random-source                                [record]

%make-random-source                            [procedure]
          (%make-random-source kind log2-period maximum-range maximum-modulus state-ref)

kind      The identifier for the random source. symbol or string

log2-period
          Period of the random source as a power of 2. fixnum

maximum-range
          Maximum range. integer

maximum-modulus
          Maximum modulus. integer

state-ref
          Procedure to return the random state. (-> random-state)

state-set!
          Procedure to set the random state. (-> random-state void)

randomize!
          Procedure to randomize the current state. (-> void)

```

```

pseudo-randomize!
    Procedure to randomize current state for substreams. (-> integer
    integer void)

make-integers
    Procedure to return a random integer stream generator. (-> (->
    (integer exact positive) (integer exact positive (<= 0 ?) (< ? ?1))))

make-reals
    Procedure to return a random inexact stream generator. (-> #!op-
    tional (real inexact (< 0 ? 1)) (-> (real inexact (< 0 ? 1))))

%random-source? [procedure]
    (%random-source? OBJECT)
    Is OBJECT a %random-source?

%random-source-kind [procedure]
    (%random-source-kind RANDOM-SOURCE)
    Returns the kind.

%random-source-log2-period [procedure]
    (%random-source-log2-period RANDOM-SOURCE)
    Returns the period base 2 exponent.

%random-source-maximum-range [procedure]
    (%random-source-maximum-range RANDOM-SOURCE)
    Returns the maximum range.

%random-source-maximum-modulus [procedure]
    (%random-source-maximum-modulus RANDOM-SOURCE)
    Returns the maximum modulus.

%random-source-state-ref [procedure]
    (%random-source-state-ref RANDOM-SOURCE)
    Returns the state-ref procedure.

%random-source-state-set! [procedure]
    (%random-source-state-set! RANDOM-SOURCE)
    Returns the state-set! procedure.

%random-source-randomize! [procedure]
    (%random-source-randomize! RANDOM-SOURCE)
    Returns the randomize! procedure.

%random-source-pseudo-randomize! [procedure]
    (%random-source-pseudo-randomize! RANDOM-SOURCE)
    Returns the pseudo-randomize! procedure.

```

<code>%make-random-source-integers</code>	[procedure]
(<code>%make-random-source-integers</code> <code>RANDOM-SOURCE</code>)	
Returns the make-integers procedure.	
<code>%make-random-source-reals</code>	[procedure]
(<code>%make-random-source-reals</code> <code>RANDOM-SOURCE</code>)	
Returns the make-reals procedure.	
<code>entropy:check-u8vector-args</code>	[procedure]
(<code>entropy:check-u8vector-args</code> <code>U8S</code> <code>U8VECTOR</code>)	
Error unless valid arguments.	
<code>entropy:check-f64vector-args</code>	[procedure]
(<code>entropy:check-f64vector-args</code> <code>F64S</code> <code>F64VECTOR</code>)	
Error unless valid arguments.	
<code>entropy-source-structure?</code>	[procedure]
(<code>entropy-source-structure?</code> <code>OBJECT</code>)	
Is <code>OBJECT</code> an entropy-source-structure?	
<code>random-source-structure?</code>	[procedure]
(<code>random-source-structure?</code> <code>OBJECT</code>)	
Is <code>OBJECT</code> a random-source-structure?	
<code>register-random-source-structure?</code>	[procedure]
(<code>register-random-source-structure?</code> <code>OBJECT</code>)	
Assuming <code>OBJECT</code> is a random-source-structure register it as a known source.	
<code>register-entropy-source-structure?</code>	[procedure]
(<code>register-entropy-source-structure?</code> <code>OBJECT</code>)	
Assuming <code>OBJECT</code> is an entropy-source-structure register it as a known source.	

3 Issues

Poor documentation.

The semi-generic nature of the random structure in the specification is odd. A random-structure must be created for every generator, but, except for 'make-random-source', the exported procedures will function with any %random-source.

The entropy structure is opposite of the random structure, mostly specific. Subject to future revision.

Random source using entropy is not provided.

4 Examples

; Please see "conf-test.scm" in this egg.

5 License

Copyright (c) 2006, 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.

Does not supercede any restrictions found in the source code.

Index

%

%entropy-source	6
%entropy-source-f64	7
%entropy-source-f64vector	7
%entropy-source-u8	7
%entropy-source-u8vector	7
%entropy-source?	7
%make-entropy-source	6
%make-random-source	7
%make-random-source-integers	9
%make-random-source-reals	9
%random-source	7
%random-source-kind	8
%random-source-log2-period	8
%random-source-maximum-modulus	8
%random-source-maximum-range	8
%random-source-pseudo-randomize!	8
%random-source-randomize!	8
%random-source-state-ref	8
%random-source-state-set!	8
%random-source?	8

C

current-entropy-source-structure	4
current-random-source-structure	4
current-seconds-entropy	3

D

dev-random-entropy	3
dev-urandom-entropy	4

E

entropy-source-signature	2
entropy-source-structure?	9
entropy-source-structures	4
entropy:check-f64vector-args	9
entropy:check-u8vector-args	9

M

make-random-bernoullis	5
make-random-binomials	5
make-random-cauchys	5
make-random-exponentials	5
make-random-gammas	5
make-random-geometrics	5
make-random-lognormals	5
make-random-normals	5
make-random-poissons	5
make-random-triangles	5
make-uniform-random-integers	4
make-uniform-random-reals	5
MRG32k3a	4
MWC	4

R

random-hollow-sphere!	6
random-normal-vector!	6
random-solid-sphere!	6
random-source-signature	2
random-source-structure?	9
random-source-structures	4
register-entropy-source-structure?	9
register-random-source-structure?	9