

Σ

A Library for ANSI Common Lisp

Christopher Mark Gore

cgore@cgore.com

<http://cgore.com/programming/lisp/sigma/>

<https://github.com/cgore/sigma>

May 3, 2013

Contents

1	Copyright	9
2	The Behave Package	11
2.1	Macros	12
2.1.1	The Behavior Macro	12
2.1.2	The Spec Macro	12
2.1.3	The Should Macro	12
2.1.4	The Should-Not Macro	12
2.1.5	The Should-Be-Null Macro	12
2.1.6	The Should-Be-A Macro	12
2.1.7	The Should= Macro	12
2.1.8	The Should/= Macro	12
2.1.9	The Should< Macro	12
2.1.10	The Should> Macro	12
2.1.11	The Should<= Macro	12
2.1.12	The Should>= Macro	12
2.1.13	The Should-Eq Macro	12
2.1.14	The Should-Not-Eq Macro	12
2.1.15	The Should-Eql Macro	12
2.1.16	The Should-Not-Eql Macro	12
2.1.17	The Should-Equal Macro	12
2.1.18	The Should-Not-Equal Macro	12
2.1.19	The Should-EqualP Macro	12
2.1.20	The Should-Not-EqualP Macro	12
2.1.21	The Should-String= Macro	12
2.1.22	The Should-Not-String= Macro	12
2.1.23	The Should-String/= Macro	12
2.1.24	The Should-Not-String/= Macro	12
2.1.25	The Should-String< Macro	12
2.1.26	The Should-Not-String< Macro	12
2.1.27	The Should-String> Macro	12
2.1.28	The Should-Not-String> Macro	12
2.1.29	The Should-String<= Macro	12
2.1.30	The Should-Not-String<= Macro	12

2.1.31	The <code>Should-String>=</code> Macro	12
2.1.32	The <code>Should-Not-String>=</code> Macro	12
2.1.33	The <code>Should-String-Equal</code> Macro	12
2.1.34	The <code>Should-Not-String-Equal</code> Macro	12
2.1.35	The <code>Should-String-Not-Equal</code> Macro	12
2.1.36	The <code>Should-Not-String-Not-Equal</code> Macro	12
2.1.37	The <code>Should-String-LessP</code> Macro	12
2.1.38	The <code>Should-Not-String-LessP</code> Macro	12
2.1.39	The <code>Should-String-GreaterP</code> Macro	12
2.1.40	The <code>Should-Not-String-GreaterP</code> Macro	12
2.1.41	The <code>Should-String-Not-GreaterP</code> Macro	12
2.1.42	The <code>Should-Not-String-Not-GreaterP</code> Macro	12
2.1.43	The <code>Should-String-Not-LessP</code> Macro	12
2.1.44	The <code>Should-Not-String-Not-LessP</code> Macro	12
3	The Control Package	13
3.1	Macros	14
3.1.1	The <code>AIIf</code> Macro	14
3.1.2	The <code>A?If</code> Macro	14
3.1.3	The <code>AAnd</code> Macro	14
3.1.4	The <code>A?And</code> Macro	14
3.1.5	The <code>ALambda</code> Macro	14
3.1.6	The <code>A?Lambda</code> Macro	14
3.1.7	The <code>ABlock</code> Macro	14
3.1.8	The <code>A?Block</code> Macro	14
3.1.9	The <code>ACond</code> Macro	14
3.1.10	The <code>A?Cond</code> Macro	14
3.1.11	The <code>AWhen</code> Macro	14
3.1.12	The <code>A?When</code> Macro	14
3.1.13	The <code>AWhile</code> Macro	14
3.1.14	The <code>A?While</code> Macro	14
3.1.15	The <code>DeleteF</code> Macro	14
3.1.16	The <code>Do-While</code> Macro	14
3.1.17	The <code>Do-Until</code> Macro	14
3.1.18	The <code>For</code> Macro	14
3.1.19	The <code>Forever</code> Macro	14
3.1.20	The <code>Multicond</code> Macro	14
3.1.21	The <code>OpF</code> Macro	14
3.1.22	The <code>Swap</code> Macro	14
3.1.23	The <code>Swap-Unless</code> Macro	14
3.1.24	The <code>Swap-When</code> Macro	14
3.1.25	The <code>Until</code> Macro	14
3.1.26	The <code>While</code> Macro	14
3.2	Functions	14
3.2.1	The <code>Compose</code> Function	14
3.2.2	The <code>Conjoin</code> Function	14

3.2.3	The Curry Function	14
3.2.4	The Disjoin Function	14
3.2.5	The Function-Alias Function	14
3.2.6	The Operator-To-Function Function	14
3.2.7	The RCompose Function	14
3.2.8	The RCurry Function	14
3.2.9	The Unimplemented Function	14
3.3	Generics	14
3.3.1	The Duplicate Generic	14
4	The Hash Package	15
4.1	Functions	15
4.1.1	The IncHash Function	15
4.1.2	The DecHash Function	15
5	The Numeric Package	17
5.1	Macros	17
5.1.1	The DivF Macro	17
5.1.2	The MultF Macro	17
5.2	Functions	17
5.2.1	The Bit? Function	17
5.2.2	The Factorial Function	17
5.2.3	The Fractional-Part Function	18
5.2.4	The Fractional-Value Function	18
5.2.5	The Integer-Range Function	18
5.2.6	The Nonnegative? Function	18
5.2.7	The Nonnegative-Integer? Function	18
5.2.8	The Positive-Integer? Function	18
5.2.9	The Product Function	18
5.2.10	The Sum Function	18
5.2.11	The Unsigned-Integer? Function	18
5.3	Types	18
5.3.1	The Nonnegative-Float Type	18
5.3.2	The Nonnegative-Integer Type	18
5.3.3	The Positive-Float Type	18
5.3.4	The Positive-Integer Type	18
6	The OS Package	19
6.1	Functions	19
6.1.1	The Perl Function	19
6.1.2	The Python Function	19
6.1.3	The Read-File Function	19
6.1.4	The Read-Lines Function	19
6.1.5	The Ruby Function	19
6.2	Parameters	19
6.2.1	The *Perl-Path* Parameter	19

6.2.2	The <code>*Python-Path*</code> Parameter	19
6.2.3	The <code>*Ruby-Path*</code> Parameter	19
7	The Probability Package	21
7.1	Macros	21
7.1.1	The <code>Decaying-Probabiliity?</code> Macro	21
7.2	Functions	21
7.2.1	The <code>Probability?</code> Function	21
7.3	Types	21
7.3.1	The <code>Probability</code> Type	21
8	The Random Package	23
8.1	Macros	23
8.1.1	The <code>NShuffle</code> Macro	23
8.2	Functions	23
8.2.1	The <code>Gauss</code> Function	23
8.2.2	The <code>Random-Argument</code> Function	23
8.2.3	The <code>Coin-Toss</code> Function	23
8.2.4	The <code>Random-In-Range</code> Function	23
8.2.5	The <code>Random-In-Ranges</code> Function	23
8.2.6	The <code>Random-Range</code> Function	23
8.2.7	The <code>Randomize-Array</code> Function	23
8.2.8	The <code>Random-Array</code> Function	23
8.3	Generics	23
8.3.1	The <code>Random-Element</code> Generic	23
8.3.2	The <code>Shuffle</code> Generic	23
9	The Sequence Package	25
9.1	Macros	26
9.1.1	The <code>Arefable?</code> Macro	26
9.1.2	The <code>NConcF</code> Macro	26
9.1.3	The <code>Nthable?</code> Macro	26
9.1.4	The <code>Set-NthCdr</code> Macro	26
9.2	Functions	26
9.2.1	The <code>Array-Values</code> Function	26
9.2.2	The <code>Nth-From-End</code> Function	26
9.2.3	The <code>Sequence?</code> Function	26
9.2.4	The <code>Empty-Sequence?</code> Function	26
9.2.5	The <code>Join-Symbol-To-All-Following</code> Function	26
9.2.6	The <code>Join-Symbol-To-All-Preceeding</code> Function	26
9.2.7	The <code>List-To-Vector</code> Function	26
9.2.8	The <code>Set-Equal</code> Function	26
9.2.9	The <code>Simple-Vector-To-List</code> Function	26
9.2.10	The <code>Sort-Order</code> Function	26
9.2.11	The <code>The-Last</code> Function	26
9.2.12	The <code>Vector-To-List</code> Function	26

9.3	Generics	26
9.3.1	The Best Generic	26
9.3.2	The Minimum Generic	26
9.3.3	The Minimum? Generic	26
9.3.4	The Maximum Generic	26
9.3.5	The Maximum? Generic	26
9.3.6	The Sort-On Generic	26
9.3.7	The Slice Generic	26
9.3.8	The Split Generic	26
9.3.9	The Worst Generic	26
10	The String Package	27
10.1	Functions	27
10.1.1	The Character-Range Function	27
10.1.2	The Character-Ranges Function	27
10.1.3	The Escape-Tildes Function	28
10.1.4	The Replace-Char Function	28
10.1.5	The StrCat Function	28
10.1.6	The StrMult Function	28
10.1.7	The String-Join Function	28
10.1.8	The Stringify Function	28
10.1.9	The To-String Function	28
10.2	Methods	28
10.2.1	The Split Methods	28
11	The Time-Series Package	29
11.1	Macros	29
11.1.1	The Snap-Index Macro	29
11.2	Functions	29
11.2.1	The Array-Raster-Line Function	29
11.2.2	The Distance Function	29
11.2.3	The Norm Function	29
11.2.4	The Raster-Line Function	29
11.2.5	The Similar-Points? Function	29
11.2.6	The Time-Series? Function	29
11.2.7	The Time-Multiseries? Function	29
11.2.8	The TMSref Function	29
11.2.9	The TMS-Dimensions Function	29
11.2.10	The TMS-Raster-Line Function	29
11.2.11	The TMS-Values Function	29
11.3	Types	29
11.3.1	The Time-Multiseries Type	29

12 The Truth Package	31
12.1 Functions	31
12.1.1 The <code>[]</code> Function	31
12.1.2 The <code>Toggle</code> Function	31
12.2 Generics	31
12.2.1 The <code>? Generic</code>	31
13 The Sigma Package	33
13.1 Variables	33
13.1.1 The <code>*Sigma-Packages*</code> Variable	33
13.2 Functions	33
13.2.1 The <code>Use-All-Sigma</code> Function	33

Chapter 1

Copyright

Copyright © 2005 – 2013, Christopher Mark Gore,
Soli Deo Gloria,
All rights reserved.

8729 Lower Marine Road, Saint Jacob, Illinois 62281 USA.

Web: <http://cgore.com>

Email: cgore@cgore.com

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Christopher Mark Gore nor the names of other contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Chapter 2

The Behave Package

2.1 Macros

2.1.1 The Behavior Macro

2.1.2 The Spec Macro

2.1.3 The Should Macro

2.1.4 The Should-Not Macro

2.1.5 The Should-Be-Null Macro

2.1.6 The Should-Be-A Macro

2.1.7 The Should= Macro

2.1.8 The Should/= Macro

2.1.9 The Should< Macro

2.1.10 The Should> Macro

2.1.11 The Should<= Macro

2.1.12 The Should>= Macro

2.1.13 The Should-Eq Macro

2.1.14 The Should-Not-Eq Macro

2.1.15 The Should-Eql Macro

2.1.16 The Should-Not-Eql Macro

2.1.17 The Should-Equal Macro

2.1.18 The Should-Not-Equal Macro

2.1.19 The Should-EqualP Macro

2.1.20 The Should-Not-EqualP Macro

2.1.21 The Should-String= Macro

2.1.22 The Should-Not-String= Macro

2.1.23 The Should-String/= Macro

Chapter 3

The Control Package

3.1 Macros

3.1.1 The AIf Macro

3.1.2 The A?If Macro

3.1.3 The AAnd Macro

3.1.4 The A?And Macro

3.1.5 The ALambda Macro

3.1.6 The A?Lambda Macro

3.1.7 The ABlock Macro

3.1.8 The A?Block Macro

3.1.9 The ACond Macro

3.1.10 The A?Cond Macro

3.1.11 The AWhen Macro

3.1.12 The A?When Macro

3.1.13 The AWhile Macro

3.1.14 The A?While Macro

3.1.15 The DeleteF Macro

3.1.16 The Do-While Macro

3.1.17 The Do-Until Macro

3.1.18 The For Macro

3.1.19 The Forever Macro

3.1.20 The Multicond Macro

3.1.21 The OpF Macro

3.1.22 The Swap Macro

3.1.23 The Swap-Unless Macro

Chapter 4

The Hash Package

4.1 Functions

4.1.1 The IncHash Function

The `IncHash` function will increment the value in *key* of the *hash*, initializing it to 1 if it isn't currently defined.

4.1.2 The DecHash Function

The `DecHash` function will decrement the value in *key* of the *hash*, initializing it to -1 if it isn't currently defined.

Chapter 5

The Numeric Package

5.1 Macros

5.1.1 The DivF Macro

5.1.2 The MultF Macro

5.2 Functions

5.2.1 The Bit? Function

5.2.2 The Factorial Function

The *Factorial* function computes $n!$ for positive integers. NB, this isn't intelligent, and uses a loop instead of better approaches.

5.2.3 The Fractional-Part Function**5.2.4 The Fractional-Value Function****5.2.5 The Integer-Range Function****5.2.6 The Nonnegative? Function****5.2.7 The Nonnegative-Integer? Function****5.2.8 The Positive-Integer? Function****5.2.9 The Product Function****5.2.10 The Sum Function****5.2.11 The Unsigned-Integer? Function****5.3 Types****5.3.1 The Nonnegative-Float Type****5.3.2 The Nonnegative-Integer Type****5.3.3 The Positive-Float Type****5.3.4 The Positive-Integer Type**

Chapter 6

The OS Package

6.1 Functions

6.1.1 The Perl Function

6.1.2 The Python Function

6.1.3 The Read-File Function

6.1.4 The Read-Lines Function

6.1.5 The Ruby Function

6.2 Parameters

6.2.1 The *Perl-Path* Parameter

6.2.2 The *Python-Path* Parameter

6.2.3 The *Ruby-Path* Parameter

Chapter 7

The Probability Package

7.1 Macros

7.1.1 The Decaying-Probabiliity? Macro

7.2 Functions

7.2.1 The Probability? Function

7.3 Types

7.3.1 The Probability Type

Chapter 8

The Random Package

8.1 Macros

8.1.1 The NShuffle Macro

8.2 Functions

8.2.1 The Gauss Function

8.2.2 The Random-Argument Function

8.2.3 The Coin-Toss Function

8.2.4 The Random-In-Range Function

8.2.5 The Random-In-Ranges Function

8.2.6 The Random-Range Function

8.2.7 The Randomize-Array Function

8.2.8 The Random-Array Function

8.3 Generics

8.3.1 The Random-Element Generic

8.3.2 The Shuffle Generic

Chapter 9

The Sequence Package

9.1 Macros

9.1.1 The Arefable? Macro

9.1.2 The NConcF Macro

9.1.3 The Nthable? Macro

9.1.4 The Set-NthCdr Macro

9.2 Functions

9.2.1 The Array-Values Function

9.2.2 The Nth-From-End Function

9.2.3 The Sequence? Function

9.2.4 The Empty-Sequence? Function

9.2.5 The Join-Symbol-To-All-Following Function

9.2.6 The Join-Symbol-To-All-Preceeding Function

9.2.7 The List-To-Vector Function

9.2.8 The Set-Equal Function

9.2.9 The Simple-Vector-To-List Function

9.2.10 The Sort-Order Function

9.2.11 The The-Last Function

9.2.12 The Vector-To-List Function

9.3 Generics

9.3.1 The Best Generic

9.3.2 The Minimum Generic

9.3.3 The Minimum? Generic

9.3.4 The Maximum Generic

Chapter 10

The String Package

The `String` package contains useful tools for working with strings.

10.1 Functions

10.1.1 The Character-Range Function

The `character-range` function returns a list of characters from the *start* to the *end* character. Note that this is returning a list, not a string.

Syntax

`(character-range start end) \implies '(start ... end)`

Arguments and Values

Start The character to start the range with, inclusive.

End The character to end the range with, inclusive.

Examples

```
(character-range #\a #\e)  $\implies$  '(\a #\b #\c #\d #\e)
(character-range #\e #\a)  $\implies$  '(\a #\b #\c #\d #\e)
```

10.1.2 The Character-Ranges Function

The `character-ranges` function is a convenience wrapper for `character-range` function, concatenating several calls and making the resultant list contain only unique instances.

Syntax

`(character-ranges start1 end1 ... \implies '(character1 ...)`

Arguments and Values

Start_n The character to start the nth range with, inclusive.

End_n The character to end the nth range with, inclusive.

Examples

`(character-ranges #\a #\c #\x #\z) \implies '(#\a #\b #\c #\x #\y #\z)`
`(character-ranges #\a #\c #\a #\c) \implies '(#\a #\b #\c)`

10.1.3 The Escape-Tildes Function**10.1.4 The Replace-Char Function****10.1.5 The StrCat Function****10.1.6 The StrMult Function****10.1.7 The String-Join Function****10.1.8 The Stringify Function****10.1.9 The To-String Function****10.2 Methods****10.2.1 The Split Methods**

Chapter 11

The Time-Series Package

11.1 Macros

11.1.1 The Snap-Index Macro

11.2 Functions

11.2.1 The Array-Raster-Line Function

11.2.2 The Distance Function

11.2.3 The Norm Function

11.2.4 The Raster-Line Function

11.2.5 The Similar-Points? Function

11.2.6 The Time-Series? Function

11.2.7 The Time-Multiseries? Function

11.2.8 The TMSref Function

11.2.9 The TMS-Dimensions Function

11.2.10 The TMS-Raster-Line Function

11.2.11 The TMS-Values Function

11.3 Types

11.3.1 The Time-Multiseries Type

Chapter 12

The Truth Package

12.1 Functions

12.1.1 The `[?]` Function

12.1.2 The `Toggle` Function

12.2 Generics

12.2.1 The `?` Generic

Chapter 13

The Sigma Package

13.1 Variables

13.1.1 The `*Sigma-Packages*` Variable

13.2 Functions

13.2.1 The `Use-All-Sigma` Function