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

Chapter 1

Copyright

8729 Lower Marine Road, Saint Jacob, Illinois 62281 USA.
Web: `http://cgore.com`
Email: `cgore@cgore.com`

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.

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

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*`)` $\Longrightarrow$ `'(`*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) ⟹ '(#\a #\b #\c #\d #\e)
(character-range #\e #\a) ⟹ '(#\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` *start*$_1$ *end*$_1$ ... $\Longrightarrow$ `'(`*character*$_1$ ...`)`

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