lfe_cl(3)

Robert Virding

2017

NAME

lfe_cl - LFE Common Lisp interface library

SYNOPSIS

This module provides a set of Common Lisp functions and macros for use in LFE. The definitions closely follow the CL definitions and won't be documented here.

DATA TYPES

The boolean values used here are the standard LFE `true` and `false` and **NOT** the Common Lisp values.

EXPORTS

Boolean conversion functions

make-lfe-bool cl-value

make-cl-bool lfe-bool

Control structures

`do vars (end-test result) body` **`[macro]`**

The value of `body` is bound the variable `do-state` which can be used when updating vars and in the `end-test`. This is the only way to get a value out of the body.

mapcar function list

maplist function list

mapc function list

mapl function list

Symbol functions

symbol-plist symbol

symbol-name symbol

get symbol pname

get symbol pname default

getl symbol pname-list

putprop symbol value pname

remprop symbol pname

Atoms (symbols) in LFE don't have property lists associated with them. However, here we have experimented with having a global ETS table `lfe-symbol-plist` which associates an atom with a property list. This is very unLFEy, but quite fun.

Property list functions

getf plist pname

getf plist pname default

putf plist value pname

remf plist pname

get-properties plist pname-list

The function `putf/3` does not exist in Common Lisp but is included to complete the operations on property lists.

Simple sequence functions

elt index sequence

length sequence

reverse sequence

Concatenation, mapping and reducing functions

some predicate sequence

every predicate sequence

notany predicate sequence

notevery predicate sequence

reduce function sequence

reduce function sequence 'initial-value x

reduce function sequence 'from-end 'true

reduce function sequence 'initial-value x 'from-end 'true

Modifying sequences

remove item sequence

remove-if predicate sequence

remove-if-not predicate sequence

remove-duplicates sequence

substitute new old sequence

substitute-if predicate sequence

substitute-if-not predicate sequence

Searching sequences

find item sequence

find-if predicate sequence

find-if-not predicate sequence

find-duplicates sequence

position item sequence

position-if predicate sequence

position-if-not predicate sequence

position-duplicates sequence

count item sequence

count-if predicate sequence

count-if-not predicate sequence

Lists

car list

first list

cdr list

rest list

nth index list

nthcdr index list

last list

butlast list

Substitution of expressions

subst new old tree

subst-if new test tree

subst-if-not new test tree

sublis alist tree

Lists as sets

member item list

member-if predicate list

member-if-not predicate list

adjoin item list

union list list

intersection list list

set-difference list list

set-exclusive-or list list

subsetp list list

Association list functions

acons key data alist

pairlis list list

pairlis list list alist

assoc key alist

assoc-if predicate alost

assoc-if-not predicate alost

rassoc key alist

rassoc-if predicate alost

rassoc-if-not predicate alost

Types

type-of object

coerce object type

Type testing macros

There is an include file which developers may which to utilize in their LFE programs: `(include-lib "lfe/include/cl.lfe")`. Currently this offers Common Lisp predicates, but may include other useful macros and functions in the future. The provided predicate macros wrap the various `is_*` Erlang functions; since these are expanded at compile time, they are usable in guards. It includes the following:

alivep x

atomp x

binaryp x

bitstringp x

boolp x

booleanp x

builtinp x

floatp x

funcp x

functionp x

intp x and **integerp x**

listp x

mapp x

numberp x

pidp x

process-alive-p x

recordp x tag

recordp x tag size

refp x

referencep x

tuplep x