Leas manual

Zach Flynn

This is the manual for Leas, the Little Extensible Accounting System, a personal account manager.

Table of Contents

1 Introduction

Leas is an interactive, command-line program for managing personal finances. Its goal is to be an extensible tool for helping you keep track of your spending and to aid in making plans. I started **Leas** to keep track of my finances leading up to my wedding. I found most free software personal finance packages were large GUI programs that had a long Time-to-Enter-Transaction: it took time to boot up an application that was not usually running and enter a transaction. This small friction made it harder for me to build the habit of keeping track of things. I also found the programs difficult to extend. So I wrote **Leas** to solve this problem. I have been using it personally since October 2018 so it works, but I'm sure there is room for improvement: make suggestions! It has a low Time-to-Enter-Transaction because you can just enter the transaction interactively at a command prompt. It is also easy to extend and script to automate different transactions or update the price of stocks, mutual funds, and ETF's. Hopefully, you will find it useful.

Leas is a command-line program, but it is possible to write a GUI on top of the program if you are looking for a project. It is possible because **Leas** can be extended using the Guile Scheme programming language. The program's prompt can execute arbitrary Scheme code. If you make a neat extension like this, let me know and I will reference it in this manual.

No knowledge of Scheme or programming in general is needed to use **Leas**. I have included almost all the functions I use personally in the distribution itself. In other words, I use it without any programming in Scheme on a day-to-day basis. The only commands I have written that are not included in the main **Leas** distribution are some commands to fetch the price of various stocks I own and update their value automatically, but I will show how to write such a command later in this document as an example.

When searching for existing Free Software solutions, I found at the other extreme programs that operated by users editing text files to add transactions (the best example being **ledger**). I found this difficult to use and wanted something more interactive. Of course, you may have your own preference. Like **ledger**, **Leas** stores data in human-readable text files. **Leas**' save files are particularly easy to analyze with statistical or spreadsheet software. The save file is simply a **tar** archive of comma-separated data files with data about your various transactions. You can open these files in **R** or **Libreoffice** and analyze your past spending that way.

All standard **Leas** commands are small Scheme functions. They use a built-in function `leas/call` which lets the user enter the function arguments interactively (it fills the role of `interactive` in Emacs, if you are familiar with it). These functions are documented in the reference manual to help you construct your own commands. The **Leas** prompt, in fact, is a Scheme interpreter. Any Scheme expression can be written there.

This document has two main sections. In the first, I describe the basic workflow for using **Leas**. This section is more in-depth than the manpage and gives some examples of how to add accounts and make transactions. It is a tutorial that will help get you up-and-running using **Leas**. In the second section, I describe all the Scheme commands exported by **Leas** which is more useful if you are trying to write your own functions.

2 Tutorial

2.1 Adding a cash asset account

A file is made up of several "accounts". To start, you will want to add some asset accounts. Assets can be wherever you store your money or your property. Asset accounts can be Checking accounts, Savings accounts, Stocks, Bonds, Cash, and anything else you own. To see how this works in **Leas**, let's look at an example of adding a checking account to a fresh file.

```
:> aa
Account: Checking
0: Expense
1: Income
2: Asset
3: Liability
Type: 2
Opening Balance: 10000
(Checking) :>
```

This opens up an asset account called "Checking" with 10000 units of currency in it as an opening balance. We can use asset accounts to pay for our expenses, to pay back our loans, and as a place to store the income we earn from working. I discuss how to do each of those things in the following sections.

2.2 Paying for expenses

Now that we have an asset account, we can use it to pay for our expenses.

First, we add an expense account, say an account for our Rent.

```
(Checking) :> aa
Account: Rent
0: Expense
1: Income
2: Asset
3: Liability
Type: 0
Opening Balance: 0
(Rent) :>
```

Now, to pay the Rent. The general "payment" command in **Leas** is the "transfer" command, `t`. `t` transfers money from one account to another account. Making a payment is transferring money from your asset accounts to your expense accounts.

```
(Rent) :> t
 0: Checking
 1: Rent
To Account: 1
 0: Checking
 1: Rent
```

```
From Account: 0
Amount: 2000
Description: The rent
Day:
Year [2019]:
Month [4]:
Day [30]:
(Checking) :>
```

You can also use the `spend` command which is like the `t` command except that it only lists assets in the *from* account and expenses in the *to* account. This is useful when you have lots of accounts. We could have done the following with an equivalent result:

```
(Rent) :> spend
 2: Rent
To Account: 2
 0: Cash
 1: Checking
From Account: 1
Amount: 2000
Description: The rent
Day:
Year [2020]:
Month [1]:
Day [22]:
(Checking) :>
```

By default, the prompt tells you what the *current account* is and, when you add an account, the current account is set to the account you just created. Several commands act on the current account. You can change the current account with the `sa` ("switch account") command,

```
(Checking) :> sa
 0: Checking
 1: Rent
Account: 1
(Rent) :>
```

The "list transactions" command is `lt`. It lists the transactions in the current account,

```
(Rent) :> lt
2019-04-30 The rent        2000.00
```

To see how much money is in your accounts, type `la` (for "list account") (`laa` lists only Asset accounts, `lal` lists only Liability accounts, and so on).

```
(Rent) :> la
Checking         8000.00    8000.00
Rent             2000.00    2000.00
```

The output has two columns. The first gives the balance in your account on the *current day*, and the second gives the balance in your account in the future.

To see the current day, use the command `cd`. To set the current day, use `sd`.

```
(Rent) :> sd
```

```
Current Day:
Year [2019]: 2018
Month [4]: 12
Day [30]: 31
(Rent) :> cd
2018-12-31
```

Now, if we type `la`, we can see what the account balance would have looked like on 2018-12-31 and what it would be in the future.

```
(Rent) :> la
Checking        10000.00    8000.00
Rent                0.00    2000.00
```

To change the day back to the current day, type `sd` and take all the default options,

```
(Rent) :> sd
Current Day:
Year [2019]:
Month [4]:
Day [30]:
(Rent) :>
```

To list only expense accounts (useful for getting an understanding of where you are spending your money),

```
(Rent) :> lae
Rent         2000.00    2000.00
```

2.3 Earning income

It is a good idea to earn income to pay for your expenses. Highly recommended. Like with paying expenses, to add income start by adding an income account. Let's start with adding a salary account,

```
(Rent) :> aa
Account: Salary
0: Expense
1: Income
2: Asset
3: Liability
Type: 1
Opening Balance: 0
(Salary) :>
```

To receive a salary, do the opposite of paying for expenses: transfer money from the salary account to an asset account.

```
(Salary) :> t
 0: Checking
 1: Rent
 2: Salary
To Account: 0
 0: Checking
```

```
 1: Rent
 2: Salary
From Account: 2
Amount: 6000
Description: Salary
Day:
Year [2019]:
Month [4]:
Day [30]:
(Salary) :> lt
2019-04-30 Salary        -6000.00
(Salary) :> la
Checking        14000.00   14000.00
Rent             2000.00    2000.00
Salary          -6000.00   -6000.00
```

2.4 Managing debt

Adding debt follows the same pattern as paying expenses and receiving income.

```
(Salary) :> aa
Account: Loan
0: Expense
1: Income
2: Asset
3: Liability
Type: 3
Opening Balance: 0
(Loan) :> t
 0: Checking
 1: Rent
 2: Salary
 3: Loan
To Account: 0
 0: Checking
 1: Rent
 2: Salary
 3: Loan
From Account: 3
Amount: 10000
Description: Personal Loan
Day:
Year [2019]:
Month [4]:
Day [30]:
(Loan) :> la
Checking        24000.00   24000.00
Rent             2000.00    2000.00
```

```
Salary          -6000.00   -6000.00
Loan           -10000.00  -10000.00
(Loan) :> lt
2019-04-30 Personal Loan       -10000.00
```

Usually, people do not give you interest-free loans. So you will also need an *expense* account for paying interest.

```
(Loan) :> aa
Account: Interest
0: Expense
1: Income
2: Asset
3: Liability
Type: 0
Opening Balance: 0
(Interest) :>
```

To pay back loans, use the command `pl`. This command allows you to split your payment on the loan between interest and principal.

```
(Interest) :> pl
 0: Checking
 1: Rent
 2: Salary
 3: Loan
 4: Interest
Loan Account: 3
 0: Checking
 1: Rent
 2: Salary
 3: Loan
 4: Interest
Interest Account: 4
 0: Checking
 1: Rent
 2: Salary
 3: Loan
 4: Interest
Pay from Account: 0
Principal: 70
Interest: 30
Description: Loan Payment
Day:
Year [2019]:
Month [4]:
Day [30]:
(Checking) :> la
Checking        23900.00   23900.00
Rent             2000.00    2000.00
```

```
Salary          -6000.00   -6000.00
Loan            -9930.00   -9930.00
Interest           30.00      30.00
```

It is useful to see broadly how much we are spending, how much we are making, and how in debt we are. To do so, we can use the command `bt`.

```
(Checking) :> bt
Expense           2030.00    2030.00
Income           -6000.00   -6000.00
Asset            23900.00   23900.00
Liability        -9930.00   -9930.00
Worth            13970.00   13970.00
Balances         10000.00
```

Note that income is measured as a *negative* number as are *liabilities*. *Worth* is *Assets* + *Liabilities* (because Liabilities are written as negative in **Leas**). Balances gives the total of the opening balances.

2.5 Non-cash assets

You may own some non-cash assets, like stocks, mutual funds, or bonds. The value of these assets in terms of currency changes over time. **Leas** provides a command for updating the currency value of these assets.

First, let's add our mutual fund,

```
(Checking) :> aa
Account: Mutual Fund
0: Expense
1: Income
2: Asset
3: Liability
Type: 2
Opening Balance: 10000
(Mutual Fund) :>
```

Then, let's add an income account for our fund.

```
(Mutual Fund) :> aa
Account: Mutual Fund Income
0: Expense
1: Income
2: Asset
3: Liability
Type: 1
Opening Balance: 0
(Mutual Fund Income) :>
```

Now, say we own 500 shares of the mutual fund each worth 20 currency units. Say the value of a share in the mutual fund increased to 21 currency units. We can then use the `csp` ("change share price") command to change the value of the shares in our mutual fund,

```
(Mutual Fund Income) :> sa
```

```
 0: Checking
 1: Rent
 2: Salary
 3: Loan
 4: Interest
 5: Mutual Fund
 6: Mutual Fund Income
Account: 5
(Mutual Fund) :> csp
 0: Checking
 1: Rent
 2: Salary
 3: Loan
 4: Interest
 5: Mutual Fund
 6: Mutual Fund Income
From Account: 6
Stock Price: 21
Number of Shares: 500
Day:
Year [2019]:
Month [4]:
Day [30]:
(Mutual Fund Income) :> lt
2019-04-30 Stock Price Change         -500.00
(Mutual Fund Income) :> la
Checking                  23900.00   23900.00
Rent                       2000.00    2000.00
Salary                    -6000.00   -6000.00
Loan                      -9930.00   -9930.00
Interest                     30.00      30.00
Mutual Fund               10500.00   10500.00
Mutual Fund Income         -500.00    -500.00
(Mutual Fund Income) :>
```

You can automate this procedure by writing a script in Scheme to fetch the new price of the fund.

2.6 Saving, loading, and quitting

To save your accounts to disk, use the `w` command.

```
(Mutual Fund Income) :> w
File: example.leas
(Mutual Fund Income) :>
```

To do so non-interactively, type `leas/write "example.leas"`.

To quit, use the `q` command,

```
(Mutual Fund Income) :> q
```

```
Save file? (yes/no) no
```

To load the file you just saved, you can use the interactive command `r`,

```
$ leas
:> r
File: example.leas
(Checking) :> la
Checking                  23900.00   23900.00
Rent                       2000.00    2000.00
Salary                    -6000.00   -6000.00
Loan                      -9930.00   -9930.00
Interest                     30.00      30.00
Mutual Fund               10500.00   10500.00
Mutual Fund Income         -500.00    -500.00
(Checking) :>
```

To load the file non-interactively, you can use the command `leas/read "example.leas"`. This command is particularly useful to include in **Leas**'s init file `~/.leasrc.scm`. Usually, this file will include the line,

```
(leas/read "/path/to/my-account.leas")
```

2.7 Scheme code

The prompt is a Scheme interpreter and can execute arbitrary code. The only difference is that the outer expression should not be enclosed in parenthesis. For example,

```
(Checking) :> begin (display (+ 1 2)) (display "\n")
3
(Checking) :>
```

Leas provides a useful function `p` for displaying expressions and then adding a newline like the above,

```
(Checking) :> p (+ 1 2)
3
(Checking) :>
```

In interactive functions, the prompt for the various arguments allows Scheme expressions as well. For example, you can enter the value of a transaction as `(- 15.29 13.99)` or `(* 0.08 123)` (for calculating a tax, for example).

2.8 Common customizations

You can add customizations that are loaded automatically in `~/.leasrc.scm`. This section gives examples for a few common customizations.

2.8.1 Set default file

You will probably want to load the same file almost everytime you load **leas**. To do so, add the following to `~/.leasrc.scm`:

```
(leas/read "/path/to/file/my-account.leas")
```

2.8.2 Customize prompt

The prompt is generated by calling the Scheme function `(leas/prompt)`. If you change this function, you can customize the prompt. For example, try adding the following to see the current date,

```
(define leas/prompt
  (lambda ()
    (if (= (leas/get-number-of-accounts) 0)
        ":> "
        (let* ((day (leas/get-current-day))
               (mday (list-ref day 0))
               (month (list-ref day 1))
               (year (list-ref day 2)))
          (string-append
           "("
           (leas/get-current-account)
           " "
           (number->string year)
           "-"
           (number->string month)
           "-"
           (number->string mday)
           ") :> "))))))
```

3 All Interactive Commands

This chapter documents all the interactive commands available in **Leas**. Its contents are similar to the manpage.

- `aa` add account
- `at` add transaction to current account
- `ltn` list latest transactions in current account
- `et X` edit transaction. X can be omitted, if it is not, display X most recent transactions instead of the default number.
- `lt` list transactions in current-account.
- `ea` edit account
- `da` delete account
- `dt` delete transaction
- `la` list accounts and their balances
- `lae` list expense accounts and their balances
- `lai` list income accounts and their balances
- `laa` list asset accounts and their balances
- `lal` list liability accounts and their balances
- `bt` current balances totaled by account type
- `cex` current total of expense accounts
- `cin` current total of income accounts
- `cas` current total of asset accounts
- `cli` current total of liability accounts
- `cwo` current assets - liabilities
- `cba` total balances
- `re` list transactions in current account matching a regular expression
- `sa` set current account
- `ca` display current account
- `w` save to file
- `r` read from file
- `dtr` delete transfer originating from a given account
- `ltbd` print transactions between two dates
- `v` display version of **Leas**
- `sd` set current day
- `cd` display current day
- `baod` find the balance of the current account over several days (ex: used for seeing weekly balances for the last month)
- `exod` see expenses over date range
- `inod` see income over date range

- `asod` see assets over date range
- `liod` see liabilities over date range
- `wood` see worth over date range
- `ttbd` see total transactions in an account between dates, over a date range
- `ttre` see total transactions in current account over a date range that match a regular expression
- `pl` pay a loan, splitting payments between interest (an expense) and the principal liability
- `fn` print current file name
- `csp` change the price of a stock in the current account
- `cal X` prints the result of the UNIX command `(string-append ‘‘cal ’’ X)`.
- `t` transfer money between two accounts
- `spend` Create a transfer from an asset account to an expense account
- `charge` Create a transfer from a liability account to an expense account
- `earn` Create a transfer from an income account to an asset account
- `borrow` Create a transfer from a liability account to an asset account.
- `q` quit
- `p X` print X, followed by a newline, for example: `p (+ 1 2)` will print `3`.

4 Leas file format

The save file is a **tar** archive containing the following files in a directory named after the save file's filename (e.g. a save file named *book.leas* would, when un-tarred, be a directory called *book*):

- accounts - a CSV file containing the metadata for each account, one line for each account. There is no header line giving column names. The fields are (in this order):
  - Account Type - one of expense, income, asset, liability.
  - Account Name - the name of the account
  - Opening Balance - the opening balance of the account.
- *account_name*.csv - for each account, there is a separate CSV file (the format of the filename for this CSV file implies that account names in leas need to be valid filenames). There is no header line giving column names. Each row in the file is a transaction. The fields are (in this order):
  - Account Name - will be the same for all transactions in the file, just the account name.
  - Amount - the amount of money added to or subtracted from the account.
  - Day - the day of the transaction (YYYY-MM-DD)
  - Description - a description of the transaction

5 Programming Leas

This chapter is a reference manual for the various Scheme functions available in **Leas**. It is now complete.

Only non-interactive functions are documented here because the interactive functions are mostly just wrappers of these that call `leas/call` to get the arguments.

5.1 Make Scheme functions interactive

- `(leas/call function-name options)`

  Calls the Scheme function with name `function-name` (a string) with arguments described by the list `options` entered interactively by the user. `options` is a list of pairs. Each element of the list has the following structure: the first element gives the "name" of the option (what **Leas** will prompt for) and the second element gives the "type" of the option: the kind of value it should contain. Both elements should be strings. The following are the current types allowed for `options`. If you use a type not in this list, whatever the user enters will be passed as-is to the function.

  - string - Pass whatever the user enters to the Scheme function as a string.
  - account - Pass the name of an account to the Scheme function. Prompts the user with a menu of accounts to select from.
  - expense_account - Pass the name of an account to the Scheme function. Prompts the user with a menu of *expense* accounts to select from.
  - income_account - Pass the name of an account to the Scheme function. Prompts the user with a menu of *income* accounts to select from.
  - asset_account - Pass the name of an account to the Scheme function. Prompts the user with a menu of *asset* accounts to select from.
  - liability_account - Pass the name of an account to the Scheme function. Prompts the user with a menu of *liability* accounts to select from.
  - pay_from_account - Pass the name of an account to the Scheme function. Prompts the user with a menu of *liability* and *asset* accounts to select from.
  - current_account - Pass the name of the current account to the Scheme function. Does not prompt the user.
  - type - Prompts the user to select an account type (asset, liability, income, expense). Passes the type as a string.
  - transaction - Prompts the user to select a transaction. Passes a pair of account number and transaction number to label the transaction to the Scheme function.
  - day - Prompts the user to select a day. Passes a three-element list elements — (day month year) — to the function.

5.2 Functions to add/edit/delete/get transactions

- `(leas/at account-name amount desc day)`

  Adds a transaction to the account with name `account-name` with the transaction amount being `amount`, description `desc`, and day given as `day` (a list with three elements, in this order: day, month, year). This is a primitive function that does not

add a counterbalancing transaction in any other account. Just adds or subtracts the amount from a certain account. It doesn't take the money from anywhere.

- `(leas/get-transactions account-name number)`

  Returns the most recent `number` transactions from `account-name`. A transaction in Scheme is a five element list with elements in this order: `(description amount year month day)`.

- `(leas/get-all-transactions account-name)`

  Returns all transactions from the account with name `account-name`.

- `(leas/get-transactions-by-regex account-name regex)`

  Returns all transactions from the account with name `account-name` where the transaction's description matches the regular expression `regex`.

- `(leas/get-transaction-by-location account-number transaction-number)`

  Return the transaction from the account numbered `account-number` with the transaction numbered `transaction-number`.

- `(leas/get-transactions-by-day account-name first-day last-day)`

  Return the transactions from account `account-name` that occurred between `first-day` and `last-day`.

- `(leas/t to-account from-account amount desc day)`

  Create a transfer from one account to another by creating offsetting transactions in both accounts.

- `(leas/dtr from-account to-location)`

  Delete a transaction that went from `from-account` to `to-location` (a pair giving account number and transaction number).

- `(leas/pay-loan loan-account interest-account from-account principal interest desc day)`

  Create a loan payment transaction with takes money from `from-account` and pays `interest` to `interest-account` and `principal` to `loan-account`. The description and day of the transaction are given by the other two arguments.

- `(leas/change-stock-price stock-account from-account stock-price number day)`

  Updates the stock price to `stock-price` of a stock account taking the money from `from-account` (usually an income account called something like "Stock Income"). The stock account is assumed to hold `number` shares. The `day` is the day of the transaction.

- `(leas/edit-transact transaction-location day amount desc)`

  Edits the transaction at `transaction-location` (a pair of account number and transaction number) to have `day`, `amount`, and `desc` set to the given values.

- `(leas/print-tscts tsct-list)`

  Prints out a list of transactions in a pretty way (well, at least, in a standard way).

5.3 Functions to add/edit/delete/get accounts

- `(leas/aa account-name type opening-balance)`

  Adds an account with name `account-name` of type `type` (one of "asset", "liability", "income", "expense") with opening balance `opening-balance`.

- `(leas/ea account-name new-account-name new-opening-balance)`

  Edits the account that currently has `account-name` to have a new name and a new opening balance.
- `(leas/da account-name)`

  Delete the account with `account-name`.
- `(leas/get-account account-name)`

  Return the account with name `account-name`. An account in Scheme is a list with four elements in this order: `(name type number-of-transactions opening-balance)`.
- `(leas/get-number-of-accounts)`

  Return the total number of accounts.
- `(leas/get-account-by-location account-number)`

  Return account at location `account-number`.
- `(leas/get-account-location account-name)`

  Return the location of the account with name `account-name`.

5.4 Functions to get/modify current day/account/file

- `(leas/get-current-file)`

  Returns the path to the current save file (the last save file loaded or written to).
- `(leas/set-current-day day)`

  Set the curent day to `day`, a list of three elements in this order: (day month year).
- `(leas/get-current-day)`

  Return a three-element list of (day month year) representing the current day.
- `(leas/get-current-account)`

  Returns the account name of the current account.
- `(leas/set-account account-name)`

  Sets the current account to `account-name`.

5.5 Total accounts

- `(leas/total-account account-name)`

  Return the sum of all transactions in `account-name`.
- `(leas/total-all-accounts)`

  Return the sum of all transactions across all accounts.
- `(leas/total-all-accounts-of-type type-number)`

  Return the sum of all transactions of a certain type (expense = 1, income = 2, asset = 4, liability = 8).
- `(leas/total-by-account-type)`

  Return totals for each account type as a list of pairs with the first element of the pair (the `car`) containing the name of the account type and the second element containing the total.

- `(leas/display-account-totals accts)`

  Display the total of a list of accounts (`accts`) in the Scheme format for an account (as returned by say `leas/get-account`.

- `(leas/current-total-of-type n)`

  Display the total for all accounts of type `n`, an integer, where 0 = expense, 1 = income, 2 = asset, 3 = income, 4 = worth, 5 = opening balances.

5.6 Repeat commands on different days

- `(leas/seq-days first-day last-day by)`

  Return a list of days starting at `first-day` and going to at most `last-day` where each element is separated by `by` days.

- `(leas/loop-days days current-day number exp)`

  A Scheme macro with executes the expression `exp` with each element of `days` being set to the current day. For elements of `days` after `number`, set the current day to `current-day` and return an empty list (i.e. stop looping after `number` days). Returns a list of pairs of the day and the results of the expression executed with that as the current day.

- `(leas/balance-account-on-days first-day last-day by account)`

  Returns a list of pairs going from `first-day` to `last-day` by `by` days where the first element of each pair is the day and the second element is the account balance on that day.

- `(leas/total-transact-in-account-between-days first-day last-day by account)`

  Return the same list of pairs where the second element of the pairs is instead the total amount of transactions between each day.

- `(leas/total-transact-in-account-re first-day last-day by account regex)`

  Return the total transactions in acccount on each day in set in the `account` that match the regular expression `regex`.

- `(leas/output-by-day day amount)`

  Display an amount on day. Used for formatting (day amount) pairs.

- `(leas/get-by-type-over-days first-day last-day by element)`

  Returns the same list of pairs where the second element of the pairs is the total transactions of the `element` number of the return value of `(leas/total-by-account-type)`.

- `(leas/get-by-type-over-days-for-type element)`

  Returns a function taking arguments `(first-day last-day by)` which calls `leas/get-by-type-over-days` fixing the `element` argument.

- `(leas/over-day-cmd element)`

  A Scheme macro which uses `leas/call` to interactively get `(first-day last-day by)` arguments for `(leas/get-by-type-over-days)` and calls it for `element`.

5.7 Set Leas parameters

- `(leas/set-select-transaction-number number)`

  Set the number of recent transactions to display when selecting a transaction for any operation.

- `leas/number-to-quick-list`

  The number of transactions to list when using `lt` and similar commands. Modify this variable to get more or less transactions. By default, it is `20`.

- `leas/prompt`

  A function that gets called (without arguments) to produce the prompt. You can modify this to whatever you would like. By default, it indicates what the current account is.

5.8 Saving and loading functions

- `(leas/write file-name)`

  Write the current **Leas** accounts to a file called `file-name`.

- `(leas/read file-name)`

  Read accounts into memory from the file named `file-name`.

5.9 Utility and interpreter functions

- `(q)`

  Quit Leas.

- `(p x)`

  Print out the object `x`. This is essentially just Guile's `display` function which also inserts a newline after the output.

- `(leas/v)`

  Return the version string for **Leas**.

- `(leas/day-from-time time)`

  Return a (day month year) list from a Scheme time object.

Appendix A GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released

under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

   If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

   If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

   It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

   You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

   A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any,

be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their

titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

   The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

   In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

   Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `https://www.gnu.org/licenses/`.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

    "Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

    "CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

    "Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

    An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

    The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with. . . Texts." line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.