

## Dynamic Ledger

Generated by Doxygen 1.8.5

Sun Dec 22 2013 16:56:16



# Contents

<b>1</b>	<b>Module Index</b>	<b>1</b>
1.1	Modules . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Module Documentation</b>	<b>7</b>
4.1	Totals Macros . . . . .	7
4.1.1	Detailed Description . . . . .	7
4.1.2	Macro Definition Documentation . . . . .	7
4.1.2.1	I_CLEARED . . . . .	7
4.1.2.2	I_NOT_THERE_YET . . . . .	7
4.1.2.3	I_OVERALL_BAL . . . . .	7
4.1.2.4	I_PENDING . . . . .	8
4.1.2.5	I_PENDING_BAL . . . . .	8
4.1.2.6	N_TOTALS . . . . .	8
4.2	Return Value Macros . . . . .	9
4.2.1	Detailed Description . . . . .	9
4.2.2	Macro Definition Documentation . . . . .	9
4.2.2.1	LFAILURE . . . . .	9
4.2.2.2	LNO . . . . .	9
4.2.2.3	LSUCCESS . . . . .	9
4.2.2.4	LYES . . . . .	9
4.2.2.5	NO_INDEX . . . . .	9
4.3	Internal Macros . . . . .	10
4.3.1	Detailed Description . . . . .	10
4.3.2	Macro Definition Documentation . . . . .	10
4.3.2.1	ENTRYSIZE . . . . .	10
4.3.2.2	EPS . . . . .	10

4.3.2.3	FILENAME_SIZE	10
4.3.2.4	LINESIZE	10
4.3.2.5	NFIELDS	10
4.4	Return Types	11
4.4.1	Detailed Description	11
4.4.2	Typedef Documentation	11
4.4.2.1	bool_t	11
4.4.2.2	color_t	11
4.4.2.3	err_t	11
4.4.2.4	index_t	11
4.5	Utility Functions	12
4.5.1	Detailed Description	13
4.5.2	Function Documentation	13
4.5.2.1	col_delim_char	13
4.5.2.2	col_delim_str	13
4.5.2.3	color	13
4.5.2.4	filled_partitions	13
4.5.2.5	input_file	14
4.5.2.6	legal_amounts	14
4.5.2.7	legal_double	14
4.5.2.8	legal_status_code	14
4.5.2.9	legal_status_codes	15
4.5.2.10	locked	15
4.5.2.11	output_file	15
4.5.2.12	qcmp	15
4.5.2.13	row_delim_char	16
4.5.2.14	row_delim_str	16
4.5.2.15	small_norm	16
4.5.2.16	space	16
4.5.2.17	str_equal	17
4.5.2.18	str_strip	17
4.5.2.19	unique	17
4.5.2.20	untotaled	17
4.5.2.21	usage	18
4.5.2.22	which	18
4.5.2.23	which_bank_total	18
4.5.2.24	which_credit_total	18
4.6	Ledger Memory Functions	20
4.6.1	Detailed Description	20
4.6.2	Function Documentation	20

4.6.2.1	<a href="#">alloc_entries</a>	20
4.6.2.2	<a href="#">alloc_totals</a>	20
4.6.2.3	<a href="#">copy_ledger</a>	21
4.6.2.4	<a href="#">free_entries</a>	21
4.6.2.5	<a href="#">free_for_retotal</a>	21
4.6.2.6	<a href="#">free_ledger</a>	21
4.6.2.7	<a href="#">get_names</a>	22
4.6.2.8	<a href="#">get_totals</a>	22
4.6.2.9	<a href="#">new_ledger</a>	22
4.7	<a href="#">Ledger Modify Functions</a>	23
4.7.1	<a href="#">Detailed Description</a>	24
4.7.2	<a href="#">Function Documentation</a>	24
4.7.2.1	<a href="#">clean</a>	24
4.7.2.2	<a href="#">condense</a>	24
4.7.2.3	<a href="#">copy_rows</a>	24
4.7.2.4	<a href="#">cut_rows</a>	25
4.7.2.5	<a href="#">edit_entry</a>	25
4.7.2.6	<a href="#">edit_entry_noretotals</a>	25
4.7.2.7	<a href="#">edit_row</a>	26
4.7.2.8	<a href="#">insert_filled_rows</a>	26
4.7.2.9	<a href="#">insert_rows</a>	26
4.7.2.10	<a href="#">map_to_coords</a>	27
4.7.2.11	<a href="#">map_to_coords_colmajor</a>	28
4.7.2.12	<a href="#">map_to_coords_rowmajor</a>	28
4.7.2.13	<a href="#">move_rows</a>	28
4.7.2.14	<a href="#">paste_rows</a>	29
4.7.2.15	<a href="#">permute_rows</a>	29
4.7.2.16	<a href="#">remove_rows</a>	29
4.7.2.17	<a href="#">rename_bank</a>	29
4.7.2.18	<a href="#">rename_credit</a>	30
4.7.2.19	<a href="#">rename_partition</a>	30
4.7.2.20	<a href="#">repartition</a>	30
4.7.2.21	<a href="#">retotal</a>	31
4.7.2.22	<a href="#">sort_by_status</a>	31
4.7.2.23	<a href="#">strip_ledger</a>	31
4.7.2.24	<a href="#">swap_rows</a>	31
4.7.2.25	<a href="#">trim_ledger</a>	32
4.7.2.26	<a href="#">unlock</a>	32
4.8	<a href="#">Ledger Input Functions</a>	33
4.8.1	<a href="#">Detailed Description</a>	33

4.8.2	Function Documentation . . . . .	33
4.8.2.1	get_entries_from_filename . . . . .	33
4.8.2.2	get_entries_from_stream . . . . .	33
4.8.2.3	get_entries_from_string . . . . .	33
4.8.2.4	get_ledger . . . . .	34
4.8.2.5	parse_char . . . . .	34
4.9	Ledger Output Functions . . . . .	35
4.9.1	Detailed Description . . . . .	35
4.9.2	Function Documentation . . . . .	35
4.9.2.1	print_ledger_to_filename . . . . .	35
4.9.2.2	print_ledger_to_stream . . . . .	35
4.9.2.3	print_ledger_to_string . . . . .	35
4.9.2.4	print_ledger_verbose . . . . .	36
4.10	Ledger Summary Functions . . . . .	37
4.10.1	Detailed Description . . . . .	37
4.10.2	Function Documentation . . . . .	37
4.10.2.1	print_summary_to_filename . . . . .	37
4.10.2.2	print_summary_to_stream . . . . .	37
4.10.2.3	print_summary_to_string . . . . .	37
4.11	Top Level Functions . . . . .	39
4.11.1	Detailed Description . . . . .	39
4.11.2	Function Documentation . . . . .	39
4.11.2.1	standalone . . . . .	39
4.12	Column Indices . . . . .	40
4.12.1	Detailed Description . . . . .	40
4.12.2	Macro Definition Documentation . . . . .	40
4.12.2.1	AMOUNT . . . . .	40
4.12.2.2	BANK . . . . .	40
4.12.2.3	CREDIT . . . . .	40
4.12.2.4	DESCRIPTION . . . . .	40
4.12.2.5	PARTITION . . . . .	40
4.12.2.6	STATUS . . . . .	40
4.13	Separator Macros . . . . .	41
4.13.1	Detailed Description . . . . .	41
4.13.2	Macro Definition Documentation . . . . .	41
4.13.2.1	COLUMN_SEPARATORS . . . . .	41
4.13.2.2	ROW_SEPARATORS . . . . .	41
4.14	Status Macros . . . . .	42
4.14.1	Detailed Description . . . . .	42
4.14.2	Macro Definition Documentation . . . . .	42

4.14.2.1	CREDIT_CHARGED	42
4.14.2.2	CREDIT_NOT_THERE_YET	42
4.14.2.3	CREDIT_PENDING	42
4.14.2.4	LOCKED	42
4.14.2.5	NOT_THERE_YET	42
4.14.2.6	PENDING	43
4.14.2.7	REMOVE	43
4.15	Printing Macros	44
4.15.1	Detailed Description	44
4.15.2	Macro Definition Documentation	44
4.15.2.1	NEGATIVE_COLOR	44
4.15.2.2	NORMAL_COLOR	44
4.15.2.3	POSITIVE_COLOR	44
4.15.2.4	PRINT_EMPTY_ACCOUNTS	44
4.15.2.5	USE_COLOR	44
4.15.2.6	ZERO_COLOR	44
<b>5</b>	<b>Class Documentation</b>	<b>45</b>
5.1	Ledger Struct Reference	45
5.1.1	Detailed Description	45
5.1.2	Member Data Documentation	45
5.1.2.1	bank_totals	45
5.1.2.2	banks	45
5.1.2.3	credit_totals	46
5.1.2.4	credits	46
5.1.2.5	entries	46
5.1.2.6	filename	46
5.1.2.7	nbanks	46
5.1.2.8	ncredits	46
5.1.2.9	npartitions	46
5.1.2.10	nrows	46
5.1.2.11	partition_totals	46
5.1.2.12	partitions	46
<b>6</b>	<b>File Documentation</b>	<b>47</b>
6.1	src/alloc_entries.c File Reference	47
6.1.1	Detailed Description	47
6.2	src/alloc_totals.c File Reference	47
6.2.1	Detailed Description	48
6.3	src/clean.c File Reference	48
6.3.1	Detailed Description	48

6.4	<a href="#">src/col_delim_char.c File Reference</a>	49
6.4.1	<a href="#">Detailed Description</a>	49
6.5	<a href="#">src/col_delim_str.c File Reference</a>	49
6.5.1	<a href="#">Detailed Description</a>	49
6.6	<a href="#">src/color.c File Reference</a>	50
6.6.1	<a href="#">Detailed Description</a>	50
6.7	<a href="#">src/condense.c File Reference</a>	50
6.7.1	<a href="#">Detailed Description</a>	50
6.8	<a href="#">src/copy_ledger.c File Reference</a>	51
6.8.1	<a href="#">Detailed Description</a>	51
6.9	<a href="#">src/copy_rows.c File Reference</a>	51
6.9.1	<a href="#">Detailed Description</a>	52
6.10	<a href="#">src/cut_rows.c File Reference</a>	52
6.10.1	<a href="#">Detailed Description</a>	52
6.11	<a href="#">src/edit_entry.c File Reference</a>	53
6.11.1	<a href="#">Detailed Description</a>	53
6.12	<a href="#">src/edit_entry_noretotals.c File Reference</a>	53
6.12.1	<a href="#">Detailed Description</a>	53
6.13	<a href="#">src/edit_row.c File Reference</a>	54
6.13.1	<a href="#">Detailed Description</a>	54
6.14	<a href="#">src/filled_partitions.c File Reference</a>	54
6.14.1	<a href="#">Detailed Description</a>	54
6.15	<a href="#">src/free_entries.c File Reference</a>	55
6.15.1	<a href="#">Detailed Description</a>	55
6.16	<a href="#">src/free_for_retotal.c File Reference</a>	55
6.16.1	<a href="#">Detailed Description</a>	56
6.17	<a href="#">src/free_ledger.c File Reference</a>	56
6.17.1	<a href="#">Detailed Description</a>	56
6.18	<a href="#">src/get_entries_from_filename.c File Reference</a>	57
6.18.1	<a href="#">Detailed Description</a>	57
6.19	<a href="#">src/get_entries_from_stream.c File Reference</a>	57
6.19.1	<a href="#">Detailed Description</a>	57
6.20	<a href="#">src/get_entries_from_string.c File Reference</a>	58
6.20.1	<a href="#">Detailed Description</a>	58
6.21	<a href="#">src/get_ledger.c File Reference</a>	58
6.21.1	<a href="#">Detailed Description</a>	58
6.22	<a href="#">src/get_names.c File Reference</a>	59
6.22.1	<a href="#">Detailed Description</a>	59
6.23	<a href="#">src/get_totals.c File Reference</a>	59
6.23.1	<a href="#">Detailed Description</a>	60



6.24	<a href="#">src/input_file.c File Reference</a>	60
6.24.1	<a href="#">Detailed Description</a>	60
6.25	<a href="#">src/insert_filled_rows.c File Reference</a>	61
6.25.1	<a href="#">Detailed Description</a>	61
6.26	<a href="#">src/insert_rows.c File Reference</a>	61
6.26.1	<a href="#">Detailed Description</a>	61
6.27	<a href="#">src/ledger.h File Reference</a>	62
6.27.1	<a href="#">Detailed Description</a>	66
6.28	<a href="#">src/legal_amounts.c File Reference</a>	66
6.28.1	<a href="#">Detailed Description</a>	66
6.29	<a href="#">src/legal_double.c File Reference</a>	67
6.29.1	<a href="#">Detailed Description</a>	67
6.30	<a href="#">src/legal_status_code.c File Reference</a>	67
6.30.1	<a href="#">Detailed Description</a>	67
6.31	<a href="#">src/legal_status_codes.c File Reference</a>	68
6.31.1	<a href="#">Detailed Description</a>	68
6.32	<a href="#">src/locked.c File Reference</a>	68
6.32.1	<a href="#">Detailed Description</a>	69
6.33	<a href="#">src/main.c File Reference</a>	69
6.33.1	<a href="#">Detailed Description</a>	69
6.33.2	<a href="#">Function Documentation</a>	69
6.33.2.1	<a href="#">main</a>	69
6.34	<a href="#">src/map_to_coords.c File Reference</a>	70
6.34.1	<a href="#">Detailed Description</a>	70
6.35	<a href="#">src/map_to_coords_colmajor.c File Reference</a>	70
6.35.1	<a href="#">Detailed Description</a>	70
6.36	<a href="#">src/map_to_coords_rowmajor.c File Reference</a>	71
6.36.1	<a href="#">Detailed Description</a>	71
6.37	<a href="#">src/move_rows.c File Reference</a>	71
6.37.1	<a href="#">Detailed Description</a>	71
6.38	<a href="#">src/new_ledger.c File Reference</a>	72
6.38.1	<a href="#">Detailed Description</a>	72
6.39	<a href="#">src/output_file.c File Reference</a>	72
6.39.1	<a href="#">Detailed Description</a>	73
6.40	<a href="#">src/parse_char.c File Reference</a>	73
6.40.1	<a href="#">Detailed Description</a>	73
6.41	<a href="#">src/paste_rows.c File Reference</a>	74
6.41.1	<a href="#">Detailed Description</a>	74
6.42	<a href="#">src/permute_rows.c File Reference</a>	74
6.42.1	<a href="#">Detailed Description</a>	74

6.43	<a href="#">src/print_ledger_to_filename.c File Reference</a>	75
6.43.1	<a href="#">Detailed Description</a>	75
6.44	<a href="#">src/print_ledger_to_stream.c File Reference</a>	75
6.44.1	<a href="#">Detailed Description</a>	75
6.45	<a href="#">src/print_ledger_to_string.c File Reference</a>	76
6.45.1	<a href="#">Detailed Description</a>	76
6.46	<a href="#">src/print_ledger_verbose.c File Reference</a>	76
6.46.1	<a href="#">Detailed Description</a>	77
6.47	<a href="#">src/print_summary_to_filename.c File Reference</a>	77
6.47.1	<a href="#">Detailed Description</a>	77
6.48	<a href="#">src/print_summary_to_stream.c File Reference</a>	78
6.48.1	<a href="#">Detailed Description</a>	78
6.49	<a href="#">src/print_summary_to_string.c File Reference</a>	78
6.49.1	<a href="#">Detailed Description</a>	78
6.50	<a href="#">src/qcmp.c File Reference</a>	79
6.50.1	<a href="#">Detailed Description</a>	79
6.51	<a href="#">src/remove_rows.c File Reference</a>	79
6.51.1	<a href="#">Detailed Description</a>	79
6.52	<a href="#">src/rename_bank.c File Reference</a>	80
6.52.1	<a href="#">Detailed Description</a>	80
6.53	<a href="#">src/rename_credit.c File Reference</a>	80
6.53.1	<a href="#">Detailed Description</a>	81
6.54	<a href="#">src/rename_partition.c File Reference</a>	81
6.54.1	<a href="#">Detailed Description</a>	81
6.55	<a href="#">src/repartition.c File Reference</a>	82
6.55.1	<a href="#">Detailed Description</a>	82
6.56	<a href="#">src/retotal.c File Reference</a>	82
6.56.1	<a href="#">Detailed Description</a>	82
6.57	<a href="#">src/row_delim_char.c File Reference</a>	83
6.57.1	<a href="#">Detailed Description</a>	83
6.58	<a href="#">src/row_delim_str.c File Reference</a>	83
6.58.1	<a href="#">Detailed Description</a>	84
6.59	<a href="#">src/small_norm.c File Reference</a>	84
6.59.1	<a href="#">Detailed Description</a>	84
6.60	<a href="#">src/sort_by_status.c File Reference</a>	84
6.60.1	<a href="#">Detailed Description</a>	85
6.61	<a href="#">src/space.c File Reference</a>	85
6.61.1	<a href="#">Detailed Description</a>	85
6.62	<a href="#">src/standalone.c File Reference</a>	86
6.62.1	<a href="#">Detailed Description</a>	86

6.63	src/str_equal.c File Reference	86
6.63.1	Detailed Description	86
6.64	src/str_strip.c File Reference	87
6.64.1	Detailed Description	87
6.65	src/strip_ledger.c File Reference	87
6.65.1	Detailed Description	87
6.66	src/swap_rows.c File Reference	88
6.66.1	Detailed Description	88
6.67	src/trim_ledger.c File Reference	88
6.67.1	Detailed Description	89
6.68	src/unique.c File Reference	89
6.68.1	Detailed Description	89
6.69	src/unlock.c File Reference	90
6.69.1	Detailed Description	90
6.70	src/untotaled.c File Reference	90
6.70.1	Detailed Description	90
6.71	src/usage.c File Reference	91
6.71.1	Detailed Description	91
6.72	src/user_settings.h File Reference	91
6.72.1	Detailed Description	92
6.73	src/which.c File Reference	93
6.73.1	Detailed Description	93
6.74	src/which_bank_total.c File Reference	93
6.74.1	Detailed Description	93
6.75	src/which_credit_total.c File Reference	94
6.75.1	Detailed Description	94
<b>Index</b>		<b>95</b>



# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

Totals Macros . . . . .	7
Return Value Macros . . . . .	9
Internal Macros . . . . .	10
Return Types . . . . .	11
Utility Functions . . . . .	12
Ledger Memory Functions . . . . .	20
Ledger Modify Functions . . . . .	23
Ledger Input Functions . . . . .	33
Ledger Output Functions . . . . .	35
Ledger Summary Functions . . . . .	37
Top Level Functions . . . . .	39
Column Indices . . . . .	40
Separator Macros . . . . .	41
Status Macros . . . . .	42
Printing Macros . . . . .	44



## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">Ledger</a>	Stores an individual ledger . . . . .	<a href="#">45</a>
------------------------	---------------------------------------	--------------------





## Chapter 3

# File Index

### 3.1 File List

Here is a list of all documented files with brief descriptions:

src/alloc_entries.c	47
src/alloc_totals.c	47
src/clean.c	48
src/col_delim_char.c	49
src/col_delim_str.c	49
src/color.c	50
src/condense.c	50
src/copy_ledger.c	51
src/copy_rows.c	51
src/cut_rows.c	52
src/edit_entry.c	53
src/edit_entry_noretotals.c	53
src/edit_row.c	54
src/filled_partitions.c	54
src/free_entries.c	55
src/free_for_retotal.c	55
src/free_ledger.c	56
src/get_entries_from_filename.c	57
src/get_entries_from_stream.c	57
src/get_entries_from_string.c	58
src/get_ledger.c	58
src/get_names.c	59
src/get_totals.c	59
src/input_file.c	60
src/insert_filled_rows.c	61
src/insert_rows.c	61
src/ledger.h	
Main header file	62
src/legal_amounts.c	66
src/legal_double.c	67
src/legal_status_code.c	67
src/legal_status_codes.c	68
src/locked.c	68
src/main.c	69
src/map_to_coords.c	70
src/map_to_coords_colmajor.c	70
src/map_to_coords_rowmajor.c	71
src/move_rows.c	71

src/new_ledger.c	72
src/output_file.c	72
src/parse_char.c	73
src/paste_rows.c	74
src/permute_rows.c	74
src/print_ledger_to_filename.c	75
src/print_ledger_to_stream.c	75
src/print_ledger_to_string.c	76
src/print_ledger_verbose.c	76
src/print_summary_to_filename.c	77
src/print_summary_to_stream.c	78
src/print_summary_to_string.c	78
src/qcmp.c	79
src/remove_rows.c	79
src/rename_bank.c	80
src/rename_credit.c	80
src/rename_partition.c	81
src/repartition.c	82
src/retotal.c	82
src/row_delim_char.c	83
src/row_delim_str.c	83
src/small_norm.c	84
src/sort_by_status.c	84
src/space.c	85
src/standalone.c	86
src/str_equal.c	86
src/str_strip.c	87
src/strip_ledger.c	87
src/swap_rows.c	88
src/trim_ledger.c	88
src/unique.c	89
src/unlock.c	90
src/untotaled.c	90
src/usage.c	91
src/user_settings.h	
Header file for user settings	91
src/which.c	93
src/which_bank_total.c	93
src/which_credit_total.c	94

## Chapter 4

# Module Documentation

### 4.1 Totals Macros

Indices for credit and bank totals.

#### Macros

- `#define I_NOT_THERE_YET 0`
- `#define I_PENDING 1`
- `#define I_CLEARED 2`
- `#define I_PENDING_BAL 3`
- `#define I_OVERALL_BAL 4`
- `#define N_TOTALS 5`

#### 4.1.1 Detailed Description

`Ledger.bank_totals` and `Ledger.credit_totals` are arrays of doubles that store various summaries of the ledger: for example, how much total money has cleared in a particular bank. These macros store indices that say which entry in `Ledger.bank_totals` and `Ledger.credit_totals` stores which total.

#### 4.1.2 Macro Definition Documentation

##### 4.1.2.1 `#define I_CLEARED 2`

In `Ledger.bank_totals`, etc., index of "available" balance.

##### 4.1.2.2 `#define I_NOT_THERE_YET 0`

In `Ledger.bank_totals`, etc., index of money not yet arrived. This does not include pending money.

##### 4.1.2.3 `#define I_OVERALL_BAL 4`

In `Ledger.bank_totals`, etc., index of overall balance. This is the true balance: "available" balance + pending money + money not yet arrived.

#### 4.1.2.4 `#define I_PENDING 1`

In [Ledger.bank\\_totals](#), etc., index of the money listed as "pending" on the bank's website.

#### 4.1.2.5 `#define I_PENDING_BAL 3`

In [Ledger.bank\\_totals](#), etc., index of pending balance (pending money + "available" balance).

#### 4.1.2.6 `#define N_TOTALS 5`

Number of entries in [Ledger.bank\\_totals](#), etc.

## 4.2 Return Value Macros

Return values from various functions.

### Macros

- `#define LSUCCESS 0`  
*Success return code.*
- `#define LFAILURE 1`  
*Failure return code.*
- `#define LNO 0`  
*"No" return code.*
- `#define LYES 1`  
*"Yes" return code.*
- `#define NO_INDEX -1`

### 4.2.1 Detailed Description

Some functions have return types of `err_t`, `bool_t`, or `index_t`. These macros define the values that these types can take on.

### 4.2.2 Macro Definition Documentation

#### 4.2.2.1 `#define LFAILURE 1`

Return value indicating a failure for functions with return type `err_t`.

#### 4.2.2.2 `#define LNO 0`

Return value indicating a "yes" for functions with return type `bool_t`.

#### 4.2.2.3 `#define LSUCCESS 0`

Return value indicating a success for functions with return type `err_t`.

#### 4.2.2.4 `#define LYES 1`

Return value indicating a "no" for functions with return type `bool_t`.

#### 4.2.2.5 `#define NO_INDEX -1`

For functions returning an `index_t` type (usually array lookup functions like which), this is the return value indicating that the index of a candidate entry was not found in the respective array.

## 4.3 Internal Macros

Internal macros.

### Macros

- `#define ENTRYSIZE 256`  
*Maximum entry size.*
- `#define EPS 0.000025`  
*"Epsilon"*
- `#define FILENAMESIZE 256`  
*File name size.*
- `#define LINESIZE 4096`  
*Line size.*
- `#define NFIELDS 6`  
*Number of fields (columns).*
- `#define NIL ""`  
*The empty string.*

### 4.3.1 Detailed Description

These macros define internal parameters such the maximum size of filenames and the number of fields (columns) in a ledger file.

### 4.3.2 Macro Definition Documentation

#### 4.3.2.1 `#define ENTRYSIZE 256`

Maximum number of characters for the entries in [Ledger.entries](#).

#### 4.3.2.2 `#define EPS 0.000025`

A small constant used to check if doubles and floats round to 0.00.

#### 4.3.2.3 `#define FILENAMESIZE 256`

Maximum number of characters for filenames

#### 4.3.2.4 `#define LINESIZE 4096`

Maximum number of characters for each line in an input ledger file.

#### 4.3.2.5 `#define NFIELDS 6`

Number of columns in the ledger. Each column represents a particular feature of a transaction (row). The fields include transaction amount, transaction status code, credit account, bank account, bank partition, and description.

## 4.4 Return Types

Return types.

### Typedefs

- typedef int `err_t`  
*Error status type.*
- typedef int `bool_t`  
*Boolean type.*
- typedef int `index_t`  
*Array index type.*
- typedef char \* `color_t`  
*Color code type.*

#### 4.4.1 Detailed Description

Defines the return types of most functions.

#### 4.4.2 Typedef Documentation

##### 4.4.2.1 typedef int `bool_t`

Can take on values LYES or LNO.

##### 4.4.2.2 typedef char\* `color_t`

Type definition for coloring output to the terminal window.

##### 4.4.2.3 typedef int `err_t`

The error status can take on values LSUCCESS and LFAILURE.

##### 4.4.2.4 typedef int `index_t`

Index in an array (used for lookup functions).

## 4.5 Utility Functions

Utility functions.

### Functions

- `bool_t col_delim_char` (char c)  
*Checks whether a character is a column separator.*
- `bool_t filled_partitions` (Ledger \*ledger, int bank)  
*Tests if any of the partitions in a given bank have nonzero balance.*
- `bool_t input_file` (char \*filename)  
*Tests if the given input file is usable.*
- `bool_t legal_double` (char \*s)  
*Checks if a character string can be converted into a meaningful floating point number.*
- `bool_t legal_amounts` (Ledger \*ledger)  
*Checks if a the AMOUNTS column in the ledger stores strings that can be converted into meaningful floating point numbers.*
- `bool_t legal_status_code` (char \*s)  
*Checks if a character string is one of the legal status codes defined in the Status\_Macros module.*
- `bool_t legal_status_codes` (Ledger \*ledger)  
*Checks if all the character strings in the STATUS column of the ledger are legal status codes as defined in the Status\_Macros module.*
- `bool_t locked` (char \*status)  
*Checks if a status code "locks" a transaction.*
- `bool_t output_file` (char \*filename)  
*Tests if the given output file is usable.*
- `bool_t row_delim_char` (char c)  
*Checks whether a character is a row separator.*
- `bool_t small_norm` (double d)  
*Checks whether a floating point number rounds to 0.00.*
- `bool_t space` (char c)  
*Checks whether a character is a whitespace character.*
- `bool_t str_equal` (const char \*s1, const char \*s2)  
*Checks whether two strings are equal.*
- `bool_t untotaled` (Ledger \*ledger)  
*Checks if account totals have been calculated for a Ledger object.*
- `color_t color` (double d, int usecolor)  
*Finds the correct color code for an amount in summary output.*
- `index_t col_delim_str` (char \*s)  
*Finds the first column delimiter in a string.*
- `index_t row_delim_str` (char \*s)  
*Finds the first row delimiter in a string.*
- `index_t which` (char \*\*s, char \*find, int n)  
*Finds an occurrence of "find" in "s".*
- `index_t which_bank_total` (char \*status)  
*Given a status code, finds the correct index in the bank\_totals array member of the Ledger type.*
- `index_t which_credit_total` (char \*status)  
*Given a status code, finds the correct index in the credit\_totals array member of the Ledger type.*
- `int qcmp` (const void \*a, const void \*b)  
*Comparison function for character strings in qsort.*
- `err_t str_strip` (char \*s)



- *Strips leading and trailing whitespace from a character string.*
- `err_t unique` (char \*\*a, int n, char \*\*\*ret, int \*nunique)  
*Finds all the unique elements in an array of character strings.*
- `err_t usage` ()  
*Prints usage details.*

### 4.5.1 Detailed Description

These are miscellaneous utility functions. Many are for checking the quality of the data.

### 4.5.2 Function Documentation

#### 4.5.2.1 `bool_t col_delim_char ( char c )`

##### Parameters

<code>c</code>	Character to check.
----------------	---------------------

##### Returns

`bool_t`: LYES or LNO

Tests if the given character is one of the legal column separators given in the character string macro, `COLUMN_SEPARATORS`.

#### 4.5.2.2 `index_t col_delim_str ( char * s )`

##### Parameters

<code>s</code>	Character string to check.
----------------	----------------------------

##### Returns

`index_t`

Loops through the characters in the argument character string and returns the index of the first character that is a column separator. if no column separator is found, the function returns `no_index`.

#### 4.5.2.3 `color_t color ( double d, int usecolor )`

##### Parameters

<code>d</code>	A double precision number representing some amount of money.
<code>usecolor</code>	An int: 1 if color-coded printing is enabled and 0 otherwise.

##### Returns

`color_t`: One of the color codes defined in the `Printing_Macros` module.

Finds the correct color code for an amount in summary output. Printing amounts in color to the terminal window makes the output prettier than otherwise. Negative amounts are printed in one color, positive amounts in another, and zeroes in a third color. Colors are defined in the `Printing_Macros` module.

#### 4.5.2.4 `bool_t filled_partitions ( Ledger * ledger, int bank )`

## Parameters

<i>ledger</i>	A pointer to a <a href="#">Ledger</a> object.
<i>bank</i>	Index of the bank in ledger->banks.

## Returns

bool\_t: LYES or LNO

Loops through all the partitions of a given bank account in a [Ledger](#) object. Returns LYES if any partition (including the unnamed partition) is found to have a nonzero amount. Returns LNO otherwise.

## 4.5.2.5 bool\_t input\_file ( char \* filename )

## Parameters

<i>filename</i>	A character array giving the full path to the filename.
-----------------	---

## Returns

bool\_t: LYES or LNO

Checks if the input file with the given full path name is usable. That is, it tries to open the file for reading and returns LYES on success and LNO on failure.

## 4.5.2.6 bool\_t legal\_amounts ( Ledger \* ledger )

## Parameters

<i>ledger</i>	A pointer to a <a href="#">Ledger</a> object.
---------------	---

## Returns

bool\_t: LYES or LNO

Loops through all the entries in the AMOUNTS column of the given [Ledger](#) object and checks that all transaction amounts (stored as human-readable character strings) can be converted into meaningful floating point numbers. Specifically, it calls legal\_double on every entry. The empty string is legal and taken to be 0.00.

## 4.5.2.7 bool\_t legal\_double ( char \* s )

## Parameters

<i>s</i>	A character string that looks like a number to humans.
----------	--

## Returns

bool\_t: LYES or LNO

Checks if a character string can be converted into a meaningful floating point number. Uses errno.h to do so.

## 4.5.2.8 bool\_t legal\_status\_code ( char \* s )

## Parameters

<i>s</i>	A character string representing a status code.
----------	--

## Returns

bool\_t: LYES or LNO

Checks if a character string is one of the legal status codes defined in the Status\_Macros module. The empty string NIL is also acceptable.

**4.5.2.9 bool\_t legal\_status\_codes ( Ledger \* ledger )**

## Parameters

<i>ledger</i>	A pointer to a <a href="#">Ledger</a> object.
---------------	---

## Returns

bool\_t: LYES or LNO

Checks if all the character strings in the STATUS column of the ledger are legal status codes as defined in the Status\_Macros module. Specifically, it loops through ledger->entries[STATUS] and calls legal\_status\_code on every entry. The empty string, NIL, is also legal.

**4.5.2.10 bool\_t locked ( char \* status )**

## Parameters

<i>status</i>	A character string representing a status code.
---------------	--

## Returns

bool\_t: LYES or LNO

Checks if a status code "locks" a transaction. Any transaction with a legal status code not equal to NIL or REMOVE is "locked": that is, it will be ignored by the trim\_ledger, condense, and clean functions.

**4.5.2.11 bool\_t output\_file ( char \* filename )**

## Parameters

<i>filename</i>	A character array giving the full path to the filename.
-----------------	---

## Returns

bool\_t: LYES or LNO

Checks if the output file with the given full path name is usable. That is, it tries to open the file for writing and returns LYES on success and LNO on failure.

**4.5.2.12 int qcmp ( const void \* a, const void \* b )**

**Parameters**

<i>a</i>	First character string.
<i>b</i>	Second character string.

**Returns**

index\_t

Comparison function for character strings in qsort. qcmp is used to sort arrays of character strings as in the function, unique.

**4.5.2.13 bool\_t row\_delim\_char ( char c )****Parameters**

<i>c</i>	Character to check.
----------	---------------------

**Returns**

bool\_t: LYES or LNO

Tests if the given character is one of the legal row separators given in the character string macro, ROW\_SEPARATORS.

**4.5.2.14 index\_t row\_delim\_str ( char \* s )****Parameters**

<i>s</i>	Character string to check.
----------	----------------------------

**Returns**

index\_t

Loops through the characters in the argument character string and returns the index of the first character that is a row separator. if no row separator is found, the function returns no\_index.

**4.5.2.15 bool\_t small\_norm ( double d )****Parameters**

<i>d</i>	Double-precision floating point number to check.
----------	--

**Returns**

bool\_t: LYES or LNO

Checks whether a floating point number rounds to 0.00. Specifically, it checks if the square of the argument is less than EPS, returns LYES if so, and LNO otherwise.

**4.5.2.16 bool\_t space ( char c )**

## Parameters

<i>c</i>	Character to check
----------	--------------------

## Returns

bool\_t: LYES or LNO

Checks whether a character is a whitespace character, returns LYES if it is, and LNO otherwise.

4.5.2.17 bool\_t str\_equal ( const char \* *s1*, const char \* *s2* )

## Parameters

<i>s1</i>	First string to check.
<i>s2</i>	Second string to check.

## Returns

bool\_t: LYES or LNO

Checks whether two strings are equal. It uses strcmp for nonnull strings, but unlike strcmp, it is safe to use with null char pointers.

4.5.2.18 err\_t str\_strip ( char \* *s* )

## Parameters

<i>s</i>	Character string
----------	------------------

## Returns

err\_t: LSUCCESS or LFAILURE

Strips leading and trailing whitespace from a character string.

4.5.2.19 err\_t unique ( char \*\* *a*, int *n*, char \*\*\* *ret*, int \* *nunique* )

## Parameters

<i>a</i>	Array of character strings.
<i>n</i>	Number of elements of <i>s</i> .
<i>ret</i>	Array of the unique elements of <i>s</i> .
<i>nunique</i>	Number elements of <i>ret</i> .

## Returns

err\_t: LSUCCESS or LFAILURE

Finds all the unique elements in an array of character strings. You can treat "a" and "n" as the arguments, and ret and nunique as the return values (passed by reference). ret and nunique should be unallocated and uninitialized when unique is called. IMPORTANT NOTE: unique adds a blank character string to the list of unique character strings if one is not already present. This is so that every ledger automatically has an unnamed bank account, an unnamed credit account, and an unnamed bank partition for every bank account.

4.5.2.20 bool\_t untotaled ( Ledger \* *ledger* )

## Parameters

<i>ledger</i>	A pointer to a <a href="#">Ledger</a> object.
---------------	---

## Returns

bool\_t: LYES or LNO

Checks if account totals have been calculated for a [Ledger](#) object. If not calculated, pointers to these totals should have already been initialized to NULL by new\_ledger. Returns LYES if totals have been calculated and LNO otherwise.

## 4.5.2.21 err\_t usage ( )

## Returns

err\_t: LSUCCESS or LFAILURE

Prints usage details of the command line interface version of the program.

## 4.5.2.22 index\_t which ( char \*\* s, char \* find, int n )

## Parameters

<i>s</i>	Array of character strings.
<i>find</i>	Character string to find.
<i>n</i>	Int: how many elements in s.

## Returns

index\_t

Finds an occurrence of the character string, "find", in the array of character strings, "s", using binary search. The index returned need not be the index of the first occurrence. Returns NO\_INDEX if "find" is not an element of s.

## 4.5.2.23 index\_t which\_bank\_total ( char \* status )

## Parameters

<i>status</i>	Character string status code.
---------------	-------------------------------

## Returns

index\_t

Given a status code, this function finds the correct index in the bank\_totals array member of the [Ledger](#) type. bank\_totals is an array indexed by the transaction status given in the Status\_Macros module. If no index is found for the given status code, NO\_INDEX is returned.

## 4.5.2.24 index\_t which\_credit\_total ( char \* status )

**Parameters**

<i>status</i>	Character string status code.
---------------	-------------------------------

**Returns**

index\_t

Given a status code, this function finds the correct index in the `credit_totals` array member of the [Ledger](#) type. `credit_totals` is an array indexed by the transaction status given in the `Status_Macros` module. If no index is found for the given status code, `NO_INDEX` is returned.

## 4.6 Ledger Memory Functions

Ledger memory functions.

### Functions

- `err_t alloc_entries (Ledger *ledger)`  
*Allocates the "entries" member of a Ledger object.*
- `err_t alloc_totals (Ledger *ledger)`  
*Allocates space for the numerical summaries of the ledger.*
- `err_t free_entries (Ledger *ledger)`  
*Frees the "entries" member of a Ledger object.*
- `err_t free_for_retotal (Ledger *ledger)`  
*Frees the account names and numerical summaries of a Ledger object.*
- `err_t free_ledger (Ledger **ledger)`  
*Frees a whole Ledger object.*
- `err_t get_names (Ledger *ledger)`  
*Gets the account names of a Ledger object.*
- `err_t get_totals (Ledger *ledger)`  
*Computes numerical summaries of a Ledger object.*
- `err_t new_ledger (Ledger **ledger)`  
*Creates a new Ledger object.*
- `err_t copy_ledger (Ledger **out_ledger, Ledger *in_ledger)`  
*Copies one Ledger object into another.*

### 4.6.1 Detailed Description

These are functions for creating, initializing, copying, and destroying Ledger objects.

### 4.6.2 Function Documentation

#### 4.6.2.1 `err_t alloc_entries ( Ledger * ledger )`

##### Parameters

<i>ledger</i>	pointer to a Ledger object
---------------	----------------------------

##### Returns

`err_t`: LSUCCESS or LFAILURE

Allocate the "entries" member of a Ledger object. "entries" stores the entries of the Ledger spreadsheet with rows representing transactions and columns representing features of the transactions.

#### 4.6.2.2 `err_t alloc_totals ( Ledger * ledger )`

##### Parameters



<i>ledger</i>	pointer to a <a href="#">Ledger</a> object
---------------	--

**Returns**

err\_t: LSUCCESS or LFAILURE

Allocates space for the numerical summaries of the ledger. These include the bank\_totals, credit\_totals, and partition\_totals members of a [Ledger](#) object.

#### 4.6.2.3 err\_t copy\_ledger ( Ledger \*\* out\_ledger, Ledger \* in\_ledger )

**Parameters**

<i>out_ledger</i>	pointer to a pointer to the output <a href="#">Ledger</a> object.
<i>in_ledger</i>	pointer to the input <a href="#">Ledger</a> object.

**Returns**

err\_t: LSUCCESS or LFAILURE

Copies one [Ledger](#) object into another. Specifically, out\_ledger is freed, and then in\_ledger is copied into it. out\_ledger should point to NULL if it is empty.

#### 4.6.2.4 err\_t free\_entries ( Ledger \* ledger )

**Parameters**

<i>ledger</i>	pointer to a <a href="#">Ledger</a> object
---------------	--

**Returns**

err\_t: LSUCCESS or LFAILURE

Frees the "entries" member of a [Ledger](#) object. "entries" stores the entries of the [Ledger](#) spreadsheet with rows representing transactions and columns representing features of the transactions.

#### 4.6.2.5 err\_t free\_for\_retotal ( Ledger \* ledger )

**Parameters**

<i>ledger</i>	pointer to a <a href="#">Ledger</a> object.
---------------	---

**Returns**

err\_t: LSUCCESS or LFAILURE

Frees the account names and numerical summaries of a [Ledger](#) object. bank\_totals, credit\_totals, partition\_totals, banks, credits, etc. are freed. This is so that the numerical summaries and account names can be recomputed after a change to the [Ledger](#) object.

#### 4.6.2.6 err\_t free\_ledger ( Ledger \*\* ledger )

**Parameters**

<i>ledger</i>	pointer to a pointer to a <a href="#">Ledger</a> object.
---------------	--

**Returns**

err\_t: LSUCCESS or LFAILURE

Frees a whole ledger object and sets the pointer to NULL. so that the program knows that it is freed.

**4.6.2.7 err\_t get\_names ( Ledger \* ledger )****Parameters**

<i>ledger</i>	pointer to a <a href="#">Ledger</a> object.
---------------	---

**Returns**

err\_t: LSUCCESS or LFAILURE

Gets the account names of a ledger object. Specifically, fills the banks, credits, and partitions member arrays of the [Ledger](#) object.

**4.6.2.8 err\_t get\_totals ( Ledger \* ledger )****Parameters**

<i>ledger</i>	pointer to a <a href="#">Ledger</a> object.
---------------	---

**Returns**

err\_t: LSUCCESS or LFAILURE

Compute numerical summaries on a [Ledger](#) object. These summaries are stored in bank\_totals, credit\_totals, and partition\_totals.

**4.6.2.9 err\_t new\_ledger ( Ledger \*\* ledger )****Parameters**

<i>ledger</i>	pointer to a pointer to a <a href="#">Ledger</a> object.
---------------	--

**Returns**

err\_t: LSUCCESS or LFAILURE

Creates a new [Ledger](#) object. Pointers are initialized to NULL so that the program knows that they do not point to any meaningful memory. ledger should point to NULL if it is empty.

## 4.7 Ledger Modify Functions

Functions for modifying [Ledger](#) objects.

### Functions

- [err\\_t clean](#) ([Ledger](#) \*ledger, int sort\_locked)  
*Cleans a ledger object.*
- [err\\_t condense](#) ([Ledger](#) \*ledger)  
*Condenses a ledger object.*
- [err\\_t copy\\_rows](#) ([Ledger](#) \*ledger, [Ledger](#) \*\*clipboard, int \*rows, int howmany)  
*Copies selected rows (transactions).*
- [err\\_t cut\\_rows](#) ([Ledger](#) \*ledger, [Ledger](#) \*\*clipboard, int \*rows, int howmany)  
*Cut selected rows (transactions).*
- [err\\_t edit\\_entry](#) ([Ledger](#) \*ledger, char \*entry, int row, int field, int append)  
*Modify a ledger entry and update the [Ledger](#) object accordingly.*
- [err\\_t edit\\_entry\\_noretal](#) ([Ledger](#) \*ledger, char \*entry, int row, int field, int append)  
*Modify a ledger entry and DO NOT update the [Ledger](#) object accordingly.*
- [err\\_t edit\\_row](#) ([Ledger](#) \*ledger, char \*\*entries, int row, int append)  
*Modify a whole row (transaction) and update the [Ledger](#) object accordingly.*
- [err\\_t map\\_to\\_coords](#) ([Ledger](#) \*ledger, char \*entry, int \*rows, int \*fields, int howmany, int append)  
*Map a character string multiple entries.*
- [err\\_t map\\_to\\_coords\\_colmajor](#) ([Ledger](#) \*ledger, char \*entry, int \*\*coords, int howmany, int append)  
*Map a character string multiple entries.*
- [err\\_t map\\_to\\_coords\\_rowmajor](#) ([Ledger](#) \*ledger, char \*entry, int \*\*coords, int howmany, int append)  
*Map a character string multiple entries.*
- [err\\_t insert\\_rows](#) ([Ledger](#) \*ledger, int row, int howmany)  
*Insert blank rows.*
- [err\\_t insert\\_filled\\_rows](#) ([Ledger](#) \*ledger, char \*\*entries, int row, int howmany)  
*Insert filled rows.*
- [err\\_t move\\_rows](#) ([Ledger](#) \*ledger, int \*rows, int nrows, int moveto)  
*Move rows to a specified location.*
- [err\\_t paste\\_rows](#) ([Ledger](#) \*ledger, [Ledger](#) \*clipboard, int where)  
*Paste rows into a specified location.*
- [err\\_t permute\\_rows](#) ([Ledger](#) \*ledger, int \*order)  
*Permute rows (transactions).*
- [err\\_t rename\\_bank](#) ([Ledger](#) \*ledger, char \*from, char \*to)  
*Safely renames a bank account.*
- [err\\_t rename\\_credit](#) ([Ledger](#) \*ledger, char \*from, char \*to)  
*Safely renames a credit account.*
- [err\\_t rename\\_partition](#) ([Ledger](#) \*ledger, char \*bank, char \*from, char \*to)  
*Safely renames a partition bank account.*
- [err\\_t remove\\_rows](#) ([Ledger](#) \*ledger)  
*Removes rows (transactions) marked for removal.*
- [err\\_t retotal](#) ([Ledger](#) \*ledger)  
*Recalculate names and totals.*
- [err\\_t trim\\_ledger](#) ([Ledger](#) \*ledger)  
*Removes blank rows.*
- [err\\_t sort\\_by\\_status](#) ([Ledger](#) \*ledger, int sort\_locked)  
*Sorts rows (transactions) by transaction status code.*

- `err_t strip_ledger (Ledger *ledger)`  
*Removes whitespace from the entries of a [Ledger](#) object.*
- `err_t swap_rows (Ledger *ledger, int row1, int row2)`  
*Interchanges two rows of a ledger object.*
- `err_t repartition (Ledger *ledger, char *bank, char **partitions, double *amounts_arg, int npartitions, int as_percentages)`  
*Repartitions a bank account.*
- `err_t unlock (Ledger *ledger)`  
*Unlock all cleared transactions.*

#### 4.7.1 Detailed Description

These are functions for modifying [Ledger](#) objects. These include (but are not limited to) functions to edit entries and rows, copy and paste rows, and trim and condense entire [Ledger](#) objects.

#### 4.7.2 Function Documentation

##### 4.7.2.1 `err_t clean ( Ledger * ledger, int sort_locked )`

###### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>sort_locked</i>	Set to 1 to bring all locked rows to the top.

###### Returns

`err_t`: LSUCCESS or LFAILURE

Cleans a ledger object. Specifically, calls condense and sort\_by\_status.

##### 4.7.2.2 `err_t condense ( Ledger * ledger )`

###### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
---------------	---

###### Returns

`err_t`: LSUCCESS or LFAILURE

Condenses a ledger object. Specifically, this function condenses all the cleared and unlocked transactions (rows) to make a smaller ledger with the same account and partition totals.

##### 4.7.2.3 `err_t copy_rows ( Ledger * ledger, Ledger ** clipboard, int * rows, int howmany )`

###### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>clipboard</i>	Pointer to a pointer to the <a href="#">Ledger</a> object to copy the rows into.

<i>rows</i>	The indices of the rows in ledger to be copied.
<i>howmany</i>	Number of elements of rows.

**Returns**

err\_t: LSUCCESS or LFAILURE

Copies selected rows (transactions). Specifically, the rows (transactions) whose indices are in "rows" are copied from "ledger" to "clipboard". The clipboard is overwritten in this function.

#### 4.7.2.4 err\_t cut\_rows ( Ledger \* ledger, Ledger \*\* clipboard, int \* rows, int howmany )

**Parameters**

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>clipboard</i>	Pointer to a pointer to the <a href="#">Ledger</a> object to copy the rows into.
<i>rows</i>	The indices of the rows in ledger to be cut.
<i>howmany</i>	Number of elements of rows.

**Returns**

err\_t: LSUCCESS or LFAILURE

Cuts the selected rows (transactions). Specifically, copy\_rows is called, and then the copied rows are removed from the original [Ledger](#) object.

#### 4.7.2.5 err\_t edit\_entry ( Ledger \* ledger, char \* entry, int row, int field, int append )

**Parameters**

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>entry</i>	New entry (character string).
<i>row</i>	The row of the entry to overwrite.
<i>field</i>	The field (column) of the entry to overwrite.
<i>append</i>	Append option.

**Returns**

err\_t: LSUCCESS or LFAILURE

Overwrite an entry of a [Ledger](#) object with a new entry. Specifically, ledger->entries[field][row] is replaced with "entry", and the other data in the [Ledger](#) object is updated with calls to get\_names and get\_totals. The append option works as in edit\_entry\_noretal

#### 4.7.2.6 err\_t edit\_entry\_noretal ( Ledger \* ledger, char \* entry, int row, int field, int append )

**Parameters**

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>entry</i>	New entry (character string).
<i>row</i>	The row of the entry to overwrite.

<i>field</i>	The field (column) of the entry to overwrite.
<i>append</i>	Append option.

**Returns**

err\_t: LSUCCESS or LFAILURE

Overwrite an entry of a [Ledger](#) object with a new entry. Specifically, ledger->entries[field][row] is replaced with "entry", and the other data in the [Ledger](#) object is NOT updated. Set "append" to 0 to overwrite each entry, 1 to append to the head of each entry, and 2 to append to the tail of each entry.

**4.7.2.7 err\_t edit\_row ( Ledger \* ledger, char \*\* entries, int row, int append )****Parameters**

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>entries</i>	New row (array of character string entries).
<i>row</i>	The row to overwrite.
<i>append</i>	Append option.

**Returns**

err\_t: LSUCCESS or LFAILURE

Overwrite a row of a [Ledger](#) object with a new row. Specifically, ledger->entries[field][row] is replaced with entries[row] for row = 0, ..., NFIELDS. The other data in the [Ledger](#) object is updated with calls to get\_names and get\_totals. APPEND works as in edit\_entry\_noretal.

**4.7.2.8 err\_t insert\_filled\_rows ( Ledger \* ledger, char \*\* entries, int row, int howmany )****Parameters**

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>entries</i>	Content of the new row to insert multiple times.
<i>row</i>	Where to insert the rows.
<i>howmany</i>	How many blank rows to insert.

**Returns**

err\_t: LSUCCESS or LFAILURE

Insert a multiple copies of a new filled row at "row" in the ledger.

**4.7.2.9 err\_t insert\_rows ( Ledger \* ledger, int row, int howmany )****Parameters**

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>row</i>	Where to insert the rows.
<i>howmany</i>	How many blank rows to insert.

**Returns**

err\_t: LSUCCESS or LFAILURE

Insert blank rows into the "entries" member array of a [Ledger](#) object.

4.7.2.10 `err_t map_to_coords ( Ledger * ledger, char * entry, int * rows, int * fields, int howmany, int append )`

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>entry</i>	Character string to map.
<i>rows</i>	Rows of entries to map to.
<i>fields</i>	Columns of entries to map to.
<i>howmany</i>	Number of elements of rows and fields.
<i>append</i>	Append option.

## Returns

err\_t: LSUCCESS or LFAILURE

Map a single character string to multiple entries of a [Ledger](#) object. Specifically, `ledger->entries[fields[i]][rows[i]]` is replaced with "entry", for i from 0 to `length(rows) - 1`. Set "append" to 0 to overwrite each entry, 1 to append to the head of each entry, and 2 to append to the tail of each entry.

4.7.2.11 `err_t map_to_coords_colmajor ( Ledger * ledger, char * entry, int ** coords, int howmany, int append )`

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>entry</i>	Character string to map.
<i>coords</i>	Coordinates of entries to map to in column major order.
<i>howmany</i>	Number of elements of rows and fields.
<i>append</i>	Append option.

## Returns

err\_t: LSUCCESS or LFAILURE

Map a single character string to multiple entries of a [Ledger](#) object. Specifically, `ledger->entries[coords[0][i]][rows[1][i]]` is replaced with "entry", for i from 0 to `length(coords) - 1`. Set "append" to 0 to overwrite each entry, 1 to append to the head of each entry, and 2 to append to the tail of each entry.

4.7.2.12 `err_t map_to_coords_rowmajor ( Ledger * ledger, char * entry, int ** coords, int howmany, int append )`

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>entry</i>	Character string to map.
<i>coords</i>	Coordinates of entries to map to in row major order.
<i>howmany</i>	Number of elements of rows and fields.
<i>append</i>	Append option.

## Returns

err\_t: LSUCCESS or LFAILURE

Map a single character string to multiple entries of a [Ledger](#) object. Specifically, `ledger->entries[coords[i][0]][rows[i][1]]` is replaced with "entry", for i from 0 to `length(coords) - 1`. Set "append" to 0 to overwrite each entry, 1 to append to the head of each entry, and 2 to append to the tail of each entry.

4.7.2.13 `err_t move_rows ( Ledger * ledger, int * rows, int nrows, int moveto )`



## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>rows</i>	Array of the indices of the rows to move.
<i>nrows</i>	Number of rows to move.
<i>moveto</i>	Where to move the rows: i.e., the destination row index.

## Returns

err\_t: LSUCCESS or LFAILURE

Move the rows (transactions) of the "entries" member array of a [Ledger](#) object to another location (row) in the same array. The workhorse of this function is permute\_rows.

## 4.7.2.14 err\_t paste\_rows ( Ledger \* ledger, Ledger \* clipboard, int where )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>clipboard</i>	Pointer to a <a href="#">Ledger</a> object containing the rows to paste.
<i>where</i>	Where to paste the rows: i.e., the destination row index.

## Returns

err\_t: LSUCCESS or LFAILURE

Pastes rows (transactions) in "clipboard" into "ledger" at row "where". insert\_rows is called, and then the relevant rows are copied in.

## 4.7.2.15 err\_t permute\_rows ( Ledger \* ledger, int \* order )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>order</i>	Array of indices to sort the rows by.

## Returns

err\_t: LSUCCESS or LFAILURE

Permute the rows (transactions) of a [Ledger](#) object. Specifically, bubble sort is applied to the int vector, "order". The same sorting operations on "order" are applied to the rows (transactions) of the [Ledger](#) object in ledger->entries.

## 4.7.2.16 err\_t remove\_rows ( Ledger \* ledger )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
---------------	---

## Returns

err\_t: LSUCCESS or LFAILURE

Remove the rows in a [Ledger](#) object with transaction status code REMOVE. The rows with this status are sent to the bottom of the "entries" array of the [Ledger](#) object and then freed.

## 4.7.2.17 err\_t rename\_bank ( Ledger \* ledger, char \* from, char \* to )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>from</i>	Which bank account to rename.
<i>to</i>	New name for the bank account.

## Returns

err\_t: LSUCCESS or LFAILURE

Safely renames a bank account and updates the other data in the [Ledger](#) object to reflect the change.

#### 4.7.2.18 err\_t rename\_credit ( Ledger \* ledger, char \* from, char \* to )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>from</i>	Which credit account to rename.
<i>to</i>	New name for the credit account.

## Returns

err\_t: LSUCCESS or LFAILURE

Safely renames a credit account and updates the other data in the [Ledger](#) object to reflect the change.

#### 4.7.2.19 err\_t rename\_partition ( Ledger \* ledger, char \* bank, char \* from, char \* to )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>bank</i>	Bank account of the partition to rename.
<i>from</i>	Which partition to rename.
<i>to</i>	New name for the partition.

## Returns

err\_t: LSUCCESS or LFAILURE

Safely renames a partition of a bank account and updates the other data in the [Ledger](#) object to reflect the change.

#### 4.7.2.20 err\_t repartition ( Ledger \* ledger, char \* bank, char \*\* partitions, double \* amounts\_arg, int npartitions, int as\_percentages )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>bank</i>	Bank account to repartition.
<i>partitions</i>	Array of character strings giving the names of the new partitions.
<i>amounts_arg</i>	Amount allocated to each partition
<i>npartitions</i>	Number of character strings in "partitions"

<i>as_percentages</i>	Whether to interpret the entries in <i>amounts_arg</i> as percentages.
-----------------------	--

## Returns

*err\_t*: LSUCCESS or LFAILURE

Repartition a bank account, allocating *amounts\_arg[i]* of money to partition *partitions[i]*. Set *as\_percentages* to 1 to interpret the elements of *amounts\_arg* as percentages, in which case the sum of the entries of *amounts\_arg* must equal 100. Set *as\_percentages* to 0 otherwise, in which case the sum of the entries of *amounts\_arg* must equal the overall true balance of the given bank account.

4.7.2.21 *err\_t* retotal ( *Ledger* \* *ledger* )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
---------------	---

## Returns

*err\_t*: LSUCCESS or LFAILURE

Call *free\_for\_retotal*, *get\_names*, and *get\_totals* to reflect any recent changes to *ledger->entries*.

4.7.2.22 *err\_t* sort\_by\_status ( *Ledger* \* *ledger*, int *sort\_locked* )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>sort_locked</i>	Set to 1 to send all locked transactions to the top and 0 otherwise.

## Returns

*err\_t*: LSUCCESS or LFAILURE

Sorts the rows (transactions) of a [Ledger](#) object such that the transactions that have not completely cleared rise to the top of the ledger. This is useful because delayed transactions have status codes that will eventually need to be changed. Calls *permute\_rows* to do the job.

4.7.2.23 *err\_t* strip\_ledger ( *Ledger* \* *ledger* )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
---------------	---

## Returns

*err\_t*: LSUCCESS or LFAILURE

Removes leading and trailing whitespace from every entry of a [Ledger](#) object. Specifically, *str\_strip* is called on every character string in the "entries" member array of the [Ledger](#) object.

4.7.2.24 *err\_t* swap\_rows ( *Ledger* \* *ledger*, int *row1*, int *row2* )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>row1</i>	Index of the first row.
<i>row2</i>	Index of the second row.

## Returns

err\_t: LSUCCESS or LFAILURE

Interchanges two rows of a [Ledger](#) object. Rows row1 and row2 of the "entries" member array of the [Ledger](#) object are interchanged.

#### 4.7.2.25 err\_t trim\_ledger ( Ledger \* ledger )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
---------------	---

## Returns

err\_t: LSUCCESS or LFAILURE

Remove rows (transactions) with transaction amounts of zero. These empty transactions do not actually contribute to the content of the ledger. They are marked for removal and then removed with `remove_rows`.

#### 4.7.2.26 err\_t unlock ( Ledger \* ledger )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
---------------	---

## Returns

err\_t: LSUCCESS or LFAILURE

Unlocks all cleared transactions. Specifically, this function changes the status of all transactions with status LOCKED to status NIL:

## 4.8 Ledger Input Functions

Functions for reading [Ledger](#) objects.

### Functions

- [err\\_t parse\\_char](#) ([Ledger](#) \*ledger, char c, int \*char\_index, int \*field, int \*row)  
*Parse a character while reading a ledger from a file.*
- [err\\_t get\\_entries\\_from\\_filename](#) ([Ledger](#) \*ledger, char \*filename)  
*Get ledger entries from a file.*
- [err\\_t get\\_entries\\_from\\_stream](#) ([Ledger](#) \*ledger, FILE \*fp)  
*Get ledger entries from a file stream.*
- [err\\_t get\\_entries\\_from\\_string](#) ([Ledger](#) \*ledger, char \*s)  
*Get ledger entries from a string.*
- [err\\_t get\\_ledger](#) ([Ledger](#) \*\*ledger, char \*filename, FILE \*fp, char \*str)  
*Recommended way to read in a [Ledger](#) object.*

### 4.8.1 Detailed Description

These are functions for reading [Ledger](#) objects from files, file streams, and character strings.

### 4.8.2 Function Documentation

#### 4.8.2.1 [err\\_t get\\_entries\\_from\\_filename](#) ( [Ledger](#) \* *ledger*, char \* *filename* )

##### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>filename</i>	Full path of the file to read from.

##### Returns

err\_t: LSUCCESS or LFAILURE

Read the entries of a ledger from a filename into a [Ledger](#) object. This is really a wrapper around [get\\_entries\\_from\\_stream](#). [get\\_entries\\_from\\_filename](#) mostly just opens the file and calls [get\\_entries\\_from\\_stream](#).

#### 4.8.2.2 [err\\_t get\\_entries\\_from\\_stream](#) ( [Ledger](#) \* *ledger*, FILE \* *fp* )

##### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>fp</i>	FILE pointer to the file stream to read from.

##### Returns

err\_t: LSUCCESS or LFAILURE

Get ledger entries from a file stream. This function first finds out how many rows there are in the input ledger and then iterates over the file stream and parses characters individually with [parse\\_char](#).

#### 4.8.2.3 [err\\_t get\\_entries\\_from\\_string](#) ( [Ledger](#) \* *ledger*, char \* *s* )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>s</i>	Character string to read from.

## Returns

err\_t: LSUCCESS or LFAILURE

Get ledger entries from a string. This function first finds out how many rows there are in the input ledger and the iterates over the string and parses characters individually with `parse_char`.

#### 4.8.2.4 err\_t get\_ledger ( Ledger \*\* ledger, char \* filename, FILE \* fp, char \* str )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>filename</i>	Full path of the file to read from.
<i>fp</i>	FILE pointer to the file stream to read from.
<i>str</i>	Character string to read from.

## Returns

err\_t: LSUCCESS or LFAILURE

This function is the recommended way to read in a [Ledger](#) object from some source. It creates a new [Ledger](#) object, reads in the entries from the specified source, and then calculates summary data on the ledger entries. To read from a filename, use the filename argument and set fp and str to NULL. To read from a file stream, use fp and set filename and str to NULL. To read from a string, use str and set filename and fp to NULL.

#### 4.8.2.5 err\_t parse\_char ( Ledger \* ledger, char c, int \* char\_index, int \* field, int \* row )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>c</i>	The character to parse.
<i>char_index</i>	The index of the character in the current ledger entry.
<i>field</i>	The field (column) of the current ledger entry being read.
<i>row</i>	The row (transaction) of the current ledger entry being read.

## Returns

err\_t: LSUCCESS or LFAILURE

Parse a character while reading a ledger from a file. Non-separators will be read into the "entries" member array of the [Ledger](#) object: specifically, a non-separator character c will be concatenated to the tail of `ledger->entries[*field]*row`. If c is a separating character like a row separator or a column separator, then a new entry will be started: that is, \*row and \*field will be changed and \*char\_index will be reset to 0.

## 4.9 Ledger Output Functions

Functions to output [Ledger](#) objects.

### Functions

- [err\\_t print\\_ledger\\_to\\_filename](#) ([Ledger](#) \*ledger, char \*filename)  
*Print ledger entries to a file.*
- [err\\_t print\\_ledger\\_to\\_stream](#) ([Ledger](#) \*ledger, FILE \*fp)  
*Print ledger entries to a file stream.*
- [err\\_t print\\_ledger\\_to\\_string](#) ([Ledger](#) \*ledger, char \*\*s)  
*Print ledger entries to a string.*
- [err\\_t print\\_ledger\\_verbose](#) ([Ledger](#) \*ledger, FILE \*fp)  
*Print out all the information on a [Ledger](#) object to a file stream.*

### 4.9.1 Detailed Description

These are functions to output [Ledger](#) objects to files, file streams, and character strings.

### 4.9.2 Function Documentation

#### 4.9.2.1 err\_t print\_ledger\_to\_filename ( [Ledger](#) \* ledger, char \* filename )

##### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>filename</i>	Full path of the file to print to.

##### Returns

err\_t: LSUCCESS or LFAILURE

Print the ledger entries of a [Ledger](#) object to a file. This function is a wrapper around `print_ledger_to_stream`. All it does is safely open the file and then call `print_ledger_to_stream`.

#### 4.9.2.2 err\_t print\_ledger\_to\_stream ( [Ledger](#) \* ledger, FILE \* fp )

##### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>fp</i>	FILE pointer to the file stream to print to.

##### Returns

err\_t: LSUCCESS or LFAILURE

Prints the ledger entries of a [Ledger](#) object to a file stream in a format that can be easily read back into a [Ledger](#) object by `get_ledger`.

#### 4.9.2.3 err\_t print\_ledger\_to\_string ( [Ledger](#) \* ledger, char \*\* s )

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>s</i>	Character string to write to.

## Returns

err\_t: LSUCCESS or LFAILURE

Prints the ledger entries of a [Ledger](#) object to a character string in a format that can be easily read back into a [Ledger](#) object by `get_ledger`.

#### 4.9.2.4 `err_t print_ledger_verbose ( Ledger * ledger, FILE * fp )`

## Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>fp</i>	FILE pointer to the file stream to print to.

## Returns

err\_t: LSUCCESS or LFAILURE

Print all the information about a [Ledger](#) object to a file stream. This function is intended for debugging purposes only. The output is ugly.



## 4.10 Ledger Summary Functions

Functions to summarize [Ledger](#) objects.

### Functions

- `err_t print_summary_to_filename (Ledger *ledger, char *filename, int usecolor)`  
Print out a summary of a [Ledger](#) object to a file.
- `err_t print_summary_to_stream (Ledger *ledger, FILE *fp, int usecolor)`  
Print out a summary of a [Ledger](#) object to a file.
- `err_t print_summary_to_string (Ledger *ledger, char **s, int usecolor)`  
Print out a summary of a [Ledger](#) object to a file.

### 4.10.1 Detailed Description

These are functions to output *summaries* of [Ledger](#) objects to files, file streams, and character strings.

### 4.10.2 Function Documentation

#### 4.10.2.1 `err_t print_summary_to_filename ( Ledger * ledger, char * filename, int usecolor )`

##### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>filename</i>	Full path of the file to print to.
<i>usecolor</i>	Include command line interface color codes?

##### Returns

`err_t`: LSUCCESS or LFAILURE

Print a summary of a [Ledger](#) object to a file. This function is a wrapper around `print_summary_to_stream`. All it does is open the file and call `print_summary_to_stream`. Set `usecolor` to 1 to print with command line interface color codes defined in the `Printing_Macros` module. Set `usecolor` to 0 to not use these color codes.

#### 4.10.2.2 `err_t print_summary_to_stream ( Ledger * ledger, FILE * fp, int usecolor )`

##### Parameters

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>fp</i>	FILE pointer of file stream to print to.
<i>usecolor</i>	Include command line interface color codes?

##### Returns

`err_t`: LSUCCESS or LFAILURE

Print a summary of a [Ledger](#) object to a file stream. Set `usecolor` to 1 to print with command line interface color codes defined in the `Printing_Macros` module. Set `usecolor` to 0 to not use these color codes.

#### 4.10.2.3 `err_t print_summary_to_string ( Ledger * ledger, char ** s, int usecolor )`

**Parameters**

<i>ledger</i>	Pointer to a <a href="#">Ledger</a> object.
<i>s</i>	Character string to print to.
<i>usecolor</i>	Include command line interface color codes?

**Returns**

err\_t: LSUCCESS or LFAILURE

Print a summary of a [Ledger](#) object to a file stream. Set usecolor to 1 to print with command line interface color codes defined in the Printing\_Macros module. Set usecolor to 0 to not use these color codes.

## 4.11 Top Level Functions

Top level functions.

### Functions

- [err\\_t standalone](#) (int argc, char \*\*argv)

*Top level function of the standalone command line interface version.*

#### 4.11.1 Detailed Description

These are functions that govern the program at the top level. For example, [standalone\(\)](#) is the main function of the command line interface version of the program.

#### 4.11.2 Function Documentation

##### 4.11.2.1 err\_t standalone ( int argc, char \*\* argv )

###### Parameters

<i>argc</i>	Number of arguments to int main.
<i>argv</i>	Arguments to int main.

###### Returns

err\_t: LSUCCESS or LFAILURE

Top level function of the standalone command line interface version of this program. Receives arguments argc and argv directly from int main. If the user calls the program with no arguments, standalone will print the usage information and return. If the user gives one argument, standalone will take the argument as the full path of a ledger file, read it in, and print it in color to the Terminal window. If two arguments are given, standalone will read a ledger from the file whose full path is given by the first argument, condense the ledger, and then write the condensed ledger to a new file whose full path is the second argument.

## 4.12 Column Indices

Ordering of the columns in the ledger file.

### Macros

- `#define AMOUNT 0`  
*Amount index.*
- `#define STATUS 1`  
*Status index.*
- `#define CREDIT 2`  
*Credit index.*
- `#define BANK 3`  
*Bank index.*
- `#define PARTITION 4`  
*Partition index.*
- `#define DESCRIPTION 5`  
*Description index.*

### 4.12.1 Detailed Description

These numbers define the ordering of the columns Each macro stores a column index. For example, if BANK is 3, then the ledger file should have bank account names in column 3. All column indices must be from 0 to 5 inclusive.

### 4.12.2 Macro Definition Documentation

#### 4.12.2.1 `#define AMOUNT 0`

Column index for transaction amounts in the ledger file.

#### 4.12.2.2 `#define BANK 3`

Column index for bank account names in the ledger file.

#### 4.12.2.3 `#define CREDIT 2`

Column index for credit account names in the ledger file.

#### 4.12.2.4 `#define DESCRIPTION 5`

Column index for transaction descriptions in the ledger file.

#### 4.12.2.5 `#define PARTITION 4`

Column index for bank partition names in the ledger file.

#### 4.12.2.6 `#define STATUS 1`

Column index for transaction status codes in the ledger file.

## 4.13 Separator Macros

Row and column separators.

### Macros

- `#define ROW_SEPARATORS "\n\r"`  
*Row separators.*
- `#define COLUMN_SEPARATORS "\t"`  
*Column separators.*

### 4.13.1 Detailed Description

These macros define which characters are used to separate rows and columns of the ledger file. For example, if `COLUMN_SEPARATORS` is `","` then the program will expect the ledger file to be stored in Comma Separated Values (CSV) format. The row separators should be different from the column separators.

### 4.13.2 Macro Definition Documentation

#### 4.13.2.1 `#define COLUMN_SEPARATORS "\t"`

Characters that encode the end of a column in the ledger file

#### 4.13.2.2 `#define ROW_SEPARATORS "\n\r"`

Characters that encode the end of a row in the ledger file

## 4.14 Status Macros

Transaction status codes.

### Macros

- `#define CREDIT_NOT_THERE_YET "cn"`  
*Made with a credit account, but not arrived online.*
- `#define CREDIT_PENDING "cp"`  
*Pending in a credit account.*
- `#define CREDIT_CHARGED "c"`  
*Charged to a credit account.*
- `#define NOT_THERE_YET "n"`  
*Not yet arrived at the bank.*
- `#define PENDING "p"`  
*Pending in a bank.*
- `#define LOCKED "l"`  
*Locked status.*
- `#define REMOVE "REMOVE"`  
*Pending removal from the ledger.*

### 4.14.1 Detailed Description

Transaction status codes tell the program where each transaction is in time: i.e., whether it's pending in the bank or it has not shown up in the credit account, etc. These macros are the character strings that the user manually enters in the STATUS column of the ledger file for each transaction.

### 4.14.2 Macro Definition Documentation

#### 4.14.2.1 `#define CREDIT_CHARGED "c"`

Transactions that show up as "cleared" in a credit account, but for which no payment from a bank has been made.

#### 4.14.2.2 `#define CREDIT_NOT_THERE_YET "cn"`

Use this status if you make a transaction with a credit account and the transaction hasn't shown up yet on the credit account's website.

#### 4.14.2.3 `#define CREDIT_PENDING "cp"`

Transactions that show up as "pending" on credit accounts

#### 4.14.2.4 `#define LOCKED "l"`

Cleared, but protected: that is, if a transaction is locked and has a nonzero amount, then the trim\_ledger, condense, and clean functions will not remove it.

#### 4.14.2.5 `#define NOT_THERE_YET "n"`

The transaction has cleared the credit account (if applicable), but the transaction or associated credit card payment hasn't shown up yet on the bank account's website.

#### 4.14.2.6 #define PENDING "p"

Transactions that show up as "pending" in bank accounts.

#### 4.14.2.7 #define REMOVE "REMOVE"

About to be removed from the ledger by `remove_rows`.

## 4.15 Printing Macros

Print formatting macros.

### Macros

- `#define PRINT_EMPTY_ACCOUNTS 0`  
*Option to print empty accounts in summaries.*
- `#define USE_COLOR 1`  
*Option to use color-coded printing in summaries.*
- `#define NORMAL_COLOR "\x1B[0m"`  
*Normal text color code.*
- `#define NEGATIVE_COLOR "\x1B[31m"`  
*Negative text color code.*
- `#define POSITIVE_COLOR "\x1B[32m"`  
*Positive text color code.*
- `#define ZERO_COLOR "\x1B[34m"`  
*Zero text color code.*

### 4.15.1 Detailed Description

These macros define settings for printing summaries of ledgers.

### 4.15.2 Macro Definition Documentation

#### 4.15.2.1 `#define NEGATIVE_COLOR "\x1B[31m"`

Terminal color code for negative totals.

#### 4.15.2.2 `#define NORMAL_COLOR "\x1B[0m"`

Terminal color code for regular text.

#### 4.15.2.3 `#define POSITIVE_COLOR "\x1B[32m"`

Terminal color code for positive totals.

#### 4.15.2.4 `#define PRINT_EMPTY_ACCOUNTS 0`

Set to 1 to include empty named accounts in summaries. Set to 0 to ignore accounts with balances of \$0.00.

#### 4.15.2.5 `#define USE_COLOR 1`

Set to 1 to enable color-coded printing to the terminal in summaries. Set to 0 for no coloring.

#### 4.15.2.6 `#define ZERO_COLOR "\x1B[34m"`

Terminal color code for empty totals.



## Chapter 5

# Class Documentation

### 5.1 Ledger Struct Reference

Stores an individual ledger.

```
#include <ledger.h>
```

#### Public Attributes

- char \* [filename](#)
- char \*\* [banks](#)
- char \*\* [credits](#)
- char \*\*\* [partitions](#)
- char \*\*\* [entries](#)
- int [nrows](#)
- int [nbanks](#)
- int [ncredits](#)
- int \* [npartitions](#)
- double \*\* [bank\\_totals](#)
- double \*\* [credit\\_totals](#)
- double \*\* [partition\\_totals](#)

#### 5.1.1 Detailed Description

Stores the spreadsheet associated with the ledger along with important summaries. This is the core data type of the program.

#### 5.1.2 Member Data Documentation

##### 5.1.2.1 double\*\* Ledger::bank\_totals

Stores how much money is in each bank account. For each bank, this includes the amount of available money, pending money, etc.

##### 5.1.2.2 char\*\* Ledger::banks

Array of names of all the bank accounts in the ledger.

#### 5.1.2.3 `double** Ledger::credit_totals`

Stores how much money is in each credit account. For each account, this includes the amount of available money, pending money, etc.

#### 5.1.2.4 `char** Ledger::credits`

Array of names of all the credit accounts in the ledger.

#### 5.1.2.5 `char*** Ledger::entries`

The matrix of actual entries in the ledger. Rows are individual transactions, and columns are fields like the amount, status, and bank of the transaction.

#### 5.1.2.6 `char* Ledger::filename`

Name of the file that the ledger came from.

#### 5.1.2.7 `int Ledger::nbanks`

Number of bank accounts (including an automatic unnamed bank account).

#### 5.1.2.8 `int Ledger::ncredits`

Number of credit accounts (including an automatic unnamed credit account).

#### 5.1.2.9 `int* Ledger::npartitions`

Number of credit accounts (including an automatic unnamed partition).

#### 5.1.2.10 `int Ledger::nrows`

Number of rows in the ledger: i.e., number of transactions.

#### 5.1.2.11 `double** Ledger::partition_totals`

Stores how much money will be in each partition of each bank account after all charges clear

#### 5.1.2.12 `char*** Ledger::partitions`

Arrays of names of all the partitions of all the bank accounts in the ledger.

The documentation for this struct was generated from the following file:

- [src/ledger.h](#)

## Chapter 6

# File Documentation

### 6.1 src/alloc\_entries.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

#### Functions

- [err\\_t alloc\\_entries](#) ([Ledger](#) \*ledger)  
*Allocates the "entries" member of a [Ledger](#) object.*

#### 6.1.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

##### Copyright

GNU General Public License 3.0

### 6.2 src/alloc\_totals.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t alloc_totals` (`Ledger *ledger`)

*Allocates space for the numerical summaries of the ledger.*

### 6.2.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.3 src/clean.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t clean` (`Ledger *ledger`, `int sort_locked`)

*Cleans a ledger object.*

### 6.3.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.4 src/col\_delim\_char.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [bool\\_t col\\_delim\\_char](#) (char c)  
*Checks whether a character is a column separator.*

#### 6.4.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

##### Copyright

GNU General Public License 3.0

## 6.5 src/col\_delim\_str.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [index\\_t col\\_delim\\_str](#) (char \*s)  
*Finds the first column delimiter in a string.*

#### 6.5.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.6 src/color.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [color\\_t color](#) (double d, int usecolor)  
*Finds the correct color code for an amount in summary output.*

### 6.6.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.7 src/condense.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t condense](#) (Ledger \*ledger)  
*Condenses a ledger object.*

### 6.7.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.8 src/copy\_ledger.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- `err_t copy_ledger (Ledger **out_ledger, Ledger *in_ledger)`  
*Copies one [Ledger](#) object into another.*

### 6.8.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.9 src/copy\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t copy_rows (Ledger *ledger, Ledger **clipboard, int *rows, int howmany)`  
*Copies selected rows (transactions).*

### 6.9.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.10 src/cut\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t cut_rows (Ledger *ledger, Ledger **clipboard, int *rows, int howmany)`  
*Cut selected rows (transactions).*

### 6.10.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0



## 6.11 src/edit\_entry.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t edit_entry` (`Ledger *ledger`, `char *entry`, `int row`, `int field`, `int append`)  
*Modify a ledger entry and update the `Ledger` object accordingly.*

#### 6.11.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

##### Copyright

GNU General Public License 3.0

## 6.12 src/edit\_entry\_norettotal.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t edit_entry_norettotal` (`Ledger *ledger`, `char *entry`, `int row`, `int field`, `int append`)  
*Modify a ledger entry and DO NOT update the `Ledger` object accordingly.*

#### 6.12.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.13 src/edit\_row.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t edit\\_row](#) ([Ledger](#) \*ledger, char \*\*entries, int row, int append)  
*Modify a whole row (transaction) and update the [Ledger](#) object accordingly.*

### 6.13.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.14 src/filled\_partitions.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [bool\\_t filled\\_partitions](#) ([Ledger](#) \*ledger, int bank)  
*Tests if any of the partitions in a given bank have nonzero balance.*

### 6.14.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.15 src/free\_entries.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [err\\_t free\\_entries](#) ([Ledger](#) \*ledger)  
*Frees the "entries" member of a [Ledger](#) object.*

### 6.15.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.16 src/free\_for\_retotal.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t free_for_retotal` (`Ledger *ledger`)

*Frees the account names and numerical summaries of a `Ledger` object.*

### 6.16.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.17 `src/free_ledger.c` File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t free_ledger` (`Ledger **ledger`)

*Frees a whole `Ledger` object.*

### 6.17.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.18 src/get\_entries\_from\_filename.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [err\\_t get\\_entries\\_from\\_filename](#) ([Ledger](#) \*ledger, char \*filename)  
*Get ledger entries from a file.*

#### 6.18.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

##### Copyright

GNU General Public License 3.0

## 6.19 src/get\_entries\_from\_stream.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [err\\_t get\\_entries\\_from\\_stream](#) ([Ledger](#) \*ledger, FILE \*fp)  
*Get ledger entries from a file stream.*

#### 6.19.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.20 src/get\_entries\_from\_string.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t get\\_entries\\_from\\_string](#) ([Ledger](#) \*ledger, char \*s)  
*Get ledger entries from a string.*

### 6.20.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.21 src/get\_ledger.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t get\\_ledger](#) ([Ledger](#) \*\*ledger, char \*filename, FILE \*fp, char \*str)  
*Recommended way to read in a [Ledger](#) object.*

### 6.21.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.22 src/get\_names.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [err\\_t get\\_names](#) ([Ledger](#) \*ledger)  
*Gets the account names of a [Ledger](#) object.*

### 6.22.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.23 src/get\_totals.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t get\\_totals](#) ([Ledger](#) \*ledger)

*Computes numerical summaries of a [Ledger](#) object.*

### 6.23.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.24 src/input\_file.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [bool\\_t input\\_file](#) (char \*filename)

*Tests if the given input file is usable.*

### 6.24.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0



## 6.25 src/insert\_filled\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [err\\_t insert\\_filled\\_rows](#) ([Ledger](#) \*ledger, char \*\*entries, int row, int howmany)  
*Insert filled rows.*

#### 6.25.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

##### Copyright

GNU General Public License 3.0

## 6.26 src/insert\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [err\\_t insert\\_rows](#) ([Ledger](#) \*ledger, int row, int howmany)  
*Insert blank rows.*

#### 6.26.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.27 src/ledger.h File Reference

Main header file.

```
#include <errno.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Classes

- struct [Ledger](#)  
*Stores an individual ledger.*

## Macros

- #define [I\\_NOT\\_THERE\\_YET](#) 0
- #define [I\\_PENDING](#) 1
- #define [I\\_CLEARED](#) 2
- #define [I\\_PENDING\\_BAL](#) 3
- #define [I\\_OVERALL\\_BAL](#) 4
- #define [N\\_TOTALS](#) 5
- #define [LSUCCESS](#) 0  
*Success return code.*
- #define [LFAILURE](#) 1  
*Failure return code.*
- #define [LNO](#) 0  
*"No" return code.*
- #define [LYES](#) 1  
*"Yes" return code.*
- #define [NO\\_INDEX](#) -1
- #define [ENTRYSIZE](#) 256  
*Maximum entry size.*
- #define [EPS](#) 0.000025  
*"Epsilon"*
- #define [FILENAME\\_SIZE](#) 256  
*File name size.*
- #define [LINESIZE](#) 4096  
*Line size.*
- #define [NFIELDS](#) 6  
*Number of fields (columns).*
- #define [NIL](#) "\0"  
*The empty string.*

## Typedefs

- typedef int [err\\_t](#)  
*Error status type.*
- typedef int [bool\\_t](#)  
*Boolean type.*
- typedef int [index\\_t](#)  
*Array index type.*
- typedef char \* [color\\_t](#)  
*Color code type.*

## Functions

- [bool\\_t col\\_delim\\_char](#) (char c)  
*Checks whether a character is a column separator.*
- [bool\\_t filled\\_partitions](#) ([Ledger](#) \*ledger, int bank)  
*Tests if any of the partitions in a given bank have nonzero balance.*
- [bool\\_t input\\_file](#) (char \*filename)  
*Tests if the given input file is usable.*
- [bool\\_t legal\\_double](#) (char \*s)  
*Checks if a character string can be converted into a meaningful floating point number.*
- [bool\\_t legal\\_amounts](#) ([Ledger](#) \*ledger)  
*Checks if a the AMOUNTS column in the ledger stores strings that can be converted into meaningful floating point numbers.*
- [bool\\_t legal\\_status\\_code](#) (char \*s)  
*Checks if a character string is one of the legal status codes defined in the Status\_Macros module.*
- [bool\\_t legal\\_status\\_codes](#) ([Ledger](#) \*ledger)  
*Checks if all the character strings in the STATUS column of the ledger are legal status codes as defined in the Status\_Macros module.*
- [bool\\_t locked](#) (char \*status)  
*Checks if a status code "locks" a transaction.*
- [bool\\_t output\\_file](#) (char \*filename)  
*Tests if the given output file is usable.*
- [bool\\_t row\\_delim\\_char](#) (char c)  
*Checks whether a character is a row separator.*
- [bool\\_t small\\_norm](#) (double d)  
*Checks whether a floating point number rounds to 0.00.*
- [bool\\_t space](#) (char c)  
*Checks whether a character is a whitespace character.*
- [bool\\_t str\\_equal](#) (const char \*s1, const char \*s2)  
*Checks whether two strings are equal.*
- [bool\\_t untotaled](#) ([Ledger](#) \*ledger)  
*Checks if account totals have been calculated for a [Ledger](#) object.*
- [color\\_t color](#) (double d, int usecolor)  
*Finds the correct color code for an amount in summary output.*
- [index\\_t col\\_delim\\_str](#) (char \*s)  
*Finds the first column delimiter in a string.*
- [index\\_t row\\_delim\\_str](#) (char \*s)  
*Finds the first row delimiter in a string.*
- [index\\_t which](#) (char \*\*s, char \*find, int n)  
*Finds an occurrence of "find" in "s".*

- [index\\_t which\\_bank\\_total](#) (char \*status)  
*Given a status code, finds the correct index in the bank\_totals array member of the [Ledger](#) type.*
- [index\\_t which\\_credit\\_total](#) (char \*status)  
*Given a status code, finds the correct index in the credit\_totals array member of the [Ledger](#) type.*
- [int qcmp](#) (const void \*a, const void \*b)  
*Comparison function for character strings in qsort.*
- [err\\_t str\\_strip](#) (char \*s)  
*Strips leading and trailing whitespace from a character string.*
- [err\\_t unique](#) (char \*\*a, int n, char \*\*\*ret, int \*nunique)  
*Finds all the unique elements in an array of character strings.*
- [err\\_t usage](#) ()  
*Prints usage details.*
- [err\\_t alloc\\_entries](#) ([Ledger](#) \*ledger)  
*Allocates the "entries" member of a [Ledger](#) object.*
- [err\\_t alloc\\_totals](#) ([Ledger](#) \*ledger)  
*Allocates space for the numerical summaries of the ledger.*
- [err\\_t free\\_entries](#) ([Ledger](#) \*ledger)  
*Frees the "entries" member of a [Ledger](#) object.*
- [err\\_t free\\_for\\_retotal](#) ([Ledger](#) \*ledger)  
*Frees the account names and numerical summaries of a [Ledger](#) object.*
- [err\\_t free\\_ledger](#) ([Ledger](#) \*\*ledger)  
*Frees a whole [Ledger](#) object.*
- [err\\_t get\\_names](#) ([Ledger](#) \*ledger)  
*Gets the account names of a [Ledger](#) object.*
- [err\\_t get\\_totals](#) ([Ledger](#) \*ledger)  
*Computes numerical summaries of a [Ledger](#) object.*
- [err\\_t new\\_ledger](#) ([Ledger](#) \*\*ledger)  
*Creates a new [Ledger](#) object.*
- [err\\_t copy\\_ledger](#) ([Ledger](#) \*\*out\_ledger, [Ledger](#) \*in\_ledger)  
*Copies one [Ledger](#) object into another.*
- [err\\_t clean](#) ([Ledger](#) \*ledger, int sort\_locked)  
*Cleans a ledger object.*
- [err\\_t condense](#) ([Ledger](#) \*ledger)  
*Condenses a ledger object.*
- [err\\_t copy\\_rows](#) ([Ledger](#) \*ledger, [Ledger](#) \*\*clipboard, int \*rows, int howmany)  
*Copies selected rows (transactions).*
- [err\\_t cut\\_rows](#) ([Ledger](#) \*ledger, [Ledger](#) \*\*clipboard, int \*rows, int howmany)  
*Cut selected rows (transactions).*
- [err\\_t edit\\_entry](#) ([Ledger](#) \*ledger, char \*entry, int row, int field, int append)  
*Modify a ledger entry and update the [Ledger](#) object accordingly.*
- [err\\_t edit\\_entry\\_norettotal](#) ([Ledger](#) \*ledger, char \*entry, int row, int field, int append)  
*Modify a ledger entry and DO NOT update the [Ledger](#) object accordingly.*
- [err\\_t edit\\_row](#) ([Ledger](#) \*ledger, char \*\*entries, int row, int append)  
*Modify a whole row (transaction) and update the [Ledger](#) object accordingly.*
- [err\\_t map\\_to\\_coords](#) ([Ledger](#) \*ledger, char \*entry, int \*rows, int \*fields, int howmany, int append)  
*Map a character string multiple entries.*
- [err\\_t map\\_to\\_coords\\_colmajor](#) ([Ledger](#) \*ledger, char \*entry, int \*\*coords, int howmany, int append)  
*Map a character string multiple entries.*
- [err\\_t map\\_to\\_coords\\_rowmajor](#) ([Ledger](#) \*ledger, char \*entry, int \*\*coords, int howmany, int append)  
*Map a character string multiple entries.*
- [err\\_t insert\\_rows](#) ([Ledger](#) \*ledger, int row, int howmany)

- Insert blank rows.*
- [err\\_t insert\\_filled\\_rows](#) ([Ledger](#) \*ledger, char \*\*entries, int row, int howmany)
- Insert filled rows.*
- [err\\_t move\\_rows](#) ([Ledger](#) \*ledger, int \*rows, int nrows, int moveto)
- Move rows to a specified location.*
- [err\\_t paste\\_rows](#) ([Ledger](#) \*ledger, [Ledger](#) \*clipboard, int where)
- Paste rows into a specified location.*
- [err\\_t permute\\_rows](#) ([Ledger](#) \*ledger, int \*order)
- Permute rows (transactions).*
- [err\\_t rename\\_bank](#) ([Ledger](#) \*ledger, char \*from, char \*to)
- Safely renames a bank account.*
- [err\\_t rename\\_credit](#) ([Ledger](#) \*ledger, char \*from, char \*to)
- Safely renames a credit account.*
- [err\\_t rename\\_partition](#) ([Ledger](#) \*ledger, char \*bank, char \*from, char \*to)
- Safely renames a partition bank account.*
- [err\\_t remove\\_rows](#) ([Ledger](#) \*ledger)
- Removes rows (transactions) marked for removal.*
- [err\\_t retotal](#) ([Ledger](#) \*ledger)
- Recalculate names and totals.*
- [err\\_t trim\\_ledger](#) ([Ledger](#) \*ledger)
- Removes blank rows.*
- [err\\_t sort\\_by\\_status](#) ([Ledger](#) \*ledger, int sort\_locked)
- Sorts rows (transactions) by transaction status code.*
- [err\\_t strip\\_ledger](#) ([Ledger](#) \*ledger)
- Removes whitespace from the entries of a [Ledger](#) object.*
- [err\\_t swap\\_rows](#) ([Ledger](#) \*ledger, int row1, int row2)
- Interchanges two rows of a ledger object.*
- [err\\_t repartition](#) ([Ledger](#) \*ledger, char \*bank, char \*\*partitions, double \*amounts\_arg, int npartitions, int as\_percentages)
- Repartitions a bank account.*
- [err\\_t unlock](#) ([Ledger](#) \*ledger)
- Unlock all cleared transactions.*
- [err\\_t parse\\_char](#) ([Ledger](#) \*ledger, char c, int \*char\_index, int \*field, int \*row)
- Parse a character while reading a ledger from a file.*
- [err\\_t get\\_entries\\_from\\_filename](#) ([Ledger](#) \*ledger, char \*filename)
- Get ledger entries from a file.*
- [err\\_t get\\_entries\\_from\\_stream](#) ([Ledger](#) \*ledger, FILE \*fp)
- Get ledger entries from a file stream.*
- [err\\_t get\\_entries\\_from\\_string](#) ([Ledger](#) \*ledger, char \*s)
- Get ledger entries from a string.*
- [err\\_t get\\_ledger](#) ([Ledger](#) \*\*ledger, char \*filename, FILE \*fp, char \*str)
- Recommended way to read in a [Ledger](#) object.*
- [err\\_t print\\_ledger\\_to\\_filename](#) ([Ledger](#) \*ledger, char \*filename)
- Print ledger entries to a file.*
- [err\\_t print\\_ledger\\_to\\_stream](#) ([Ledger](#) \*ledger, FILE \*fp)
- Print ledger entries to a file stream.*
- [err\\_t print\\_ledger\\_to\\_string](#) ([Ledger](#) \*ledger, char \*\*s)
- Print ledger entries to a string.*
- [err\\_t print\\_ledger\\_verbose](#) ([Ledger](#) \*ledger, FILE \*fp)
- Print out all the information on a [Ledger](#) object to a file stream.*
- [err\\_t print\\_summary\\_to\\_filename](#) ([Ledger](#) \*ledger, char \*filename, int usecolor)

- *Print out a summary of a [Ledger](#) object to a file.*
- [err\\_t print\\_summary\\_to\\_stream](#) ([Ledger](#) \*ledger, FILE \*fp, int usecolor)  
*Print out a summary of a [Ledger](#) object to a file.*
- [err\\_t print\\_summary\\_to\\_string](#) ([Ledger](#) \*ledger, char \*\*s, int usecolor)  
*Print out a summary of a [Ledger](#) object to a file.*
- [err\\_t standalone](#) (int argc, char \*\*argv)  
*Top level function of the standalone command line interface version.*

### 6.27.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

Includes all typedefs and function declarations, along with some of the macros.

## 6.28 src/legal\_amounts.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [bool\\_t legal\\_amounts](#) ([Ledger](#) \*ledger)  
*Checks if a the AMOUNTS column in the ledger stores strings that can be converted into meaningful floating point numbers.*

### 6.28.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.29 src/legal\_double.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `bool_t legal_double` (char \*s)

*Checks if a character string can be converted into a meaningful floating point number.*

### 6.29.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.30 src/legal\_status\_code.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `bool_t legal_status_code` (char \*s)

*Checks if a character string is one of the legal status codes defined in the Status\_Macros module.*

### 6.30.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.31 src/legal\_status\_codes.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [bool\\_t legal\\_status\\_codes](#) (Ledger \*ledger)

*Checks if all the character strings in the STATUS column of the ledger are legal status codes as defined in the Status\_Macros module.*

### 6.31.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.32 src/locked.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [bool\\_t locked](#) (char \*status)

*Checks if a status code "locks" a transaction.*



### 6.32.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.33 src/main.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 6.33.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

### 6.33.2 Function Documentation

#### 6.33.2.1 int main ( int *argc*, char \*\* *argv* )

Main function of the standalone command line interface version. Just calls standalone and returns.

## 6.34 src/map\_to\_coords.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t map_to_coords` (`Ledger *ledger`, `char *entry`, `int *rows`, `int *fields`, `int howmany`, `int append`)  
*Map a character string multiple entries.*

### 6.34.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.35 src/map\_to\_coords\_colmajor.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t map_to_coords_colmajor` (`Ledger *ledger`, `char *entry`, `int **coords`, `int howmany`, `int append`)  
*Map a character string multiple entries.*

### 6.35.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.36 src/map\_to\_coords\_rowmajor.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t map\\_to\\_coords\\_rowmajor](#) ([Ledger](#) \*ledger, char \*entry, int \*\*coords, int howmany, int append)  
*Map a character string multiple entries.*

### 6.36.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.37 src/move\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t move\\_rows](#) ([Ledger](#) \*ledger, int \*rows, int nrows, int moveto)  
*Move rows to a specified location.*

### 6.37.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.38 src/new\_ledger.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [err\\_t new\\_ledger](#) ([Ledger](#) \*\*ledger)  
*Creates a new [Ledger](#) object.*

### 6.38.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.39 src/output\_file.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `bool_t output_file` (char \*filename)  
*Tests if the given output file is usable.*

### 6.39.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.40 src/parse\_char.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t parse_char` (`Ledger` \*ledger, char c, int \*char\_index, int \*field, int \*row)  
*Parse a character while reading a ledger from a file.*

### 6.40.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.41 src/paste\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t paste_rows` (`Ledger *ledger`, `Ledger *clipboard`, `int where`)  
*Paste rows into a specified location.*

### 6.41.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.42 src/permute\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t permute_rows` (`Ledger *ledger`, `int *order`)  
*Permute rows (transactions).*

### 6.42.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.43 src/print\_ledger\_to\_filename.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t print\\_ledger\\_to\\_filename](#) ([Ledger](#) \*ledger, char \*filename)  
*Print ledger entries to a file.*

### 6.43.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.44 src/print\_ledger\_to\_stream.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t print\\_ledger\\_to\\_stream](#) ([Ledger](#) \*ledger, FILE \*fp)  
*Print ledger entries to a file stream.*

### 6.44.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.45 src/print\_ledger\_to\_string.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [err\\_t print\\_ledger\\_to\\_string](#) ([Ledger](#) \*ledger, char \*\*s)  
*Print ledger entries to a string.*

### 6.45.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.46 src/print\_ledger\_verbose.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```



## Functions

- `err_t print_ledger_verbose` (`Ledger *ledger`, `FILE *fp`)  
*Print out all the information on a `Ledger` object to a file stream.*

### 6.46.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.47 src/print\_summary\_to\_filename.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t print_summary_to_filename` (`Ledger *ledger`, `char *filename`, `int usecolor`)  
*Print out a summary of a `Ledger` object to a file.*

### 6.47.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.48 src/print\_summary\_to\_stream.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t print_summary_to_stream` (`Ledger *ledger`, `FILE *fp`, `int usecolor`)  
*Print out a summary of a `Ledger` object to a file.*

### 6.48.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.49 src/print\_summary\_to\_string.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t print_summary_to_string` (`Ledger *ledger`, `char **s`, `int usecolor`)  
*Print out a summary of a `Ledger` object to a file.*

### 6.49.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.50 src/qcmp.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `int qcmp` (const void \*a, const void \*b)  
*Comparison function for character strings in qsort.*

### 6.50.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License  $\geq$  3.0 (See COPYING.txt)

## 6.51 src/remove\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t remove_rows` (Ledger \*ledger)  
*Removes rows (transactions) marked for removal.*

### 6.51.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.52 src/rename\_bank.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [err\\_t rename\\_bank](#) ([Ledger](#) \*ledger, char \*from, char \*to)  
*Safely renames a bank account.*

### 6.52.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.53 src/rename\_credit.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t rename_credit` (`Ledger` \*ledger, char \*from, char \*to)

*Safely renames a credit account.*

### 6.53.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.54 src/rename\_partition.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t rename_partition` (`Ledger` \*ledger, char \*bank, char \*from, char \*to)

*Safely renames a partition bank account.*

### 6.54.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.55 src/repartition.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t repartition` (`Ledger *ledger`, `char *bank`, `char **partitions`, `double *amounts_arg`, `int npartitions`, `int as_percentages`)

*Repartitions a bank account.*

### 6.55.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.56 src/retotal.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t retotal` (`Ledger *ledger`)

*Recalculate names and totals.*

### 6.56.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.57 src/row\_delim\_char.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [bool\\_t row\\_delim\\_char](#) (char c)  
*Checks whether a character is a row separator.*

### 6.57.1 Detailed Description

**Author**Will Landau (<http://www.will-landau.com/>)**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.58 src/row\_delim\_str.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [index\\_t row\\_delim\\_str](#) (char \*s)  
*Finds the first row delimiter in a string.*

### 6.58.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.59 src/small\_norm.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [bool\\_t small\\_norm](#) (double d)  
*Checks whether a floating point number rounds to 0.00.*

### 6.59.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.60 src/sort\_by\_status.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```



## Functions

- `err_t sort_by_status` (`Ledger *ledger`, `int sort_locked`)  
*Sorts rows (transactions) by transaction status code.*

### 6.60.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.61 src/space.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `bool_t space` (`char c`)  
*Checks whether a character is a whitespace character.*

### 6.61.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.62 src/standalone.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [err\\_t standalone](#) (int argc, char \*\*argv)  
*Top level function of the standalone command line interface version.*

### 6.62.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.63 src/str\_equal.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [bool\\_t str\\_equal](#) (const char \*s1, const char \*s2)  
*Checks whether two strings are equal.*

### 6.63.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.64 src/str\_strip.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t str\\_strip](#) (char \*s)  
*Strips leading and trailing whitespace from a character string.*

### 6.64.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.65 src/strip\_ledger.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- [err\\_t strip\\_ledger](#) (Ledger \*ledger)  
*Removes whitespace from the entries of a [Ledger](#) object.*

### 6.65.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.66 src/swap\_rows.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [err\\_t swap\\_rows](#) ([Ledger](#) \*ledger, int row1, int row2)  
*Interchanges two rows of a ledger object.*

### 6.66.1 Detailed Description

**Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

## 6.67 src/trim\_ledger.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t trim_ledger` (`Ledger *ledger`)

*Removes blank rows.*

### 6.67.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.68 src/unique.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t unique` (`char **a`, `int n`, `char ***ret`, `int *nunique`)

*Finds all the unique elements in an array of character strings.*

### 6.68.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.69 src/unlock.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `err_t unlock (Ledger *ledger)`  
*Unlock all cleared transactions.*

### 6.69.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.70 src/untotaled.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- `bool_t untotaled (Ledger *ledger)`  
*Checks if account totals have been calculated for a [Ledger](#) object.*

### 6.70.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

## Copyright

GNU General Public License 3.0

## 6.71 src/usage.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

## Functions

- `err_t usage()`  
*Prints usage details.*

### 6.71.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License 3.0

## 6.72 src/user\_settings.h File Reference

Header file for user settings.

## Macros

- `#define AMOUNT 0`  
*Amount index.*
- `#define STATUS 1`  
*Status index.*
- `#define CREDIT 2`  
*Credit index.*
- `#define BANK 3`  
*Bank index.*
- `#define PARTITION 4`  
*Partition index.*
- `#define DESCRIPTION 5`  
*Description index.*

- `#define ROW_SEPARATORS "\n\r"`  
*Row separators.*
- `#define COLUMN_SEPARATORS "\t"`  
*Column separators.*
- `#define CREDIT_NOT_THERE_YET "cn"`  
*Made with a credit account, but not arrived online.*
- `#define CREDIT_PENDING "cp"`  
*Pending in a credit account.*
- `#define CREDIT_CHARGED "c"`  
*Charged to a credit account.*
- `#define NOT_THERE_YET "n"`  
*Not yet arrived at the bank.*
- `#define PENDING "p"`  
*Pending in a bank.*
- `#define LOCKED "l"`  
*Locked status.*
- `#define REMOVE "REMOVE"`  
*Pending removal from the ledger.*
- `#define PRINT_EMPTY_ACCOUNTS 0`  
*Option to print empty accounts in summaries.*
- `#define USE_COLOR 1`  
*Option to use color-coded printing in summaries.*
- `#define NORMAL_COLOR "\x1B[0m"`  
*Normal text color code.*
- `#define NEGATIVE_COLOR "\x1B[31m"`  
*Negative text color code.*
- `#define POSITIVE_COLOR "\x1B[32m"`  
*Positive text color code.*
- `#define ZERO_COLOR "\x1B[34m"`  
*Zero text color code.*

### 6.72.1 Detailed Description

#### Author

Will Landau (<http://www.will-landau.com/>)

#### Date

2013-2014

#### Copyright

GNU General Public License >= 3.0 (See COPYING.txt)

Users can redefine the macros in this file to set program preferences at compile time.



## 6.73 src/which.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [index\\_t which](#) (char \*\*s, char \*find, int n)  
*Finds an occurrence of "find" in "s".*

#### 6.73.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

##### Copyright

GNU General Public License 3.0

## 6.74 src/which\_bank\_total.c File Reference

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

### Functions

- [index\\_t which\\_bank\\_total](#) (char \*status)  
*Given a status code, finds the correct index in the bank\_totals array member of the [Ledger](#) type.*

#### 6.74.1 Detailed Description

##### Author

Will Landau (<http://www.will-landau.com/>)

##### Date

2013-2014

**Copyright**

GNU General Public License 3.0

**6.75 src/which\_credit\_total.c File Reference**

```
#include <errno.h>
#include "ledger.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "user_settings.h"
```

**Functions**

- [index\\_t which\\_credit\\_total](#) (char \*status)

*Given a status code, finds the correct index in the credit\_totals array member of the [Ledger](#) type.*

**6.75.1 Detailed Description****Author**

Will Landau (<http://www.will-landau.com/>)

**Date**

2013-2014

**Copyright**

GNU General Public License 3.0

# Index

- AMOUNT
  - Column Indices, [40](#)
- alloc\_entries
  - Ledger Memory Functions, [20](#)
- alloc\_totals
  - Ledger Memory Functions, [20](#)
- BANK
  - Column Indices, [40](#)
- bank\_totals
  - Ledger, [45](#)
- banks
  - Ledger, [45](#)
- bool\_t
  - Return Types, [11](#)
- COLUMN\_SEPARATORS
  - Separator Macros, [41](#)
- CREDIT
  - Column Indices, [40](#)
- CREDIT\_CHARGED
  - Status Macros, [42](#)
- CREDIT\_PENDING
  - Status Macros, [42](#)
- clean
  - Ledger Modify Functions, [24](#)
- col\_delim\_char
  - Utility Functions, [13](#)
- col\_delim\_str
  - Utility Functions, [13](#)
- color
  - Utility Functions, [13](#)
- color\_t
  - Return Types, [11](#)
- Column Indices, [40](#)
  - AMOUNT, [40](#)
  - BANK, [40](#)
  - CREDIT, [40](#)
  - DESCRIPTION, [40](#)
  - PARTITION, [40](#)
  - STATUS, [40](#)
- condense
  - Ledger Modify Functions, [24](#)
- copy\_ledger
  - Ledger Memory Functions, [21](#)
- copy\_rows
  - Ledger Modify Functions, [24](#)
- credit\_totals
  - Ledger, [45](#)
- credits
  - Ledger, [46](#)
- cut\_rows
  - Ledger Modify Functions, [25](#)
- DESCRIPTION
  - Column Indices, [40](#)
- ENTRYSIZE
  - Internal Macros, [10](#)
- EPS
  - Internal Macros, [10](#)
- edit\_entry
  - Ledger Modify Functions, [25](#)
- edit\_entry\_noretal
  - Ledger Modify Functions, [25](#)
- edit\_row
  - Ledger Modify Functions, [26](#)
- entries
  - Ledger, [46](#)
- err\_t
  - Return Types, [11](#)
- FILENAME\_SIZE
  - Internal Macros, [10](#)
- filename
  - Ledger, [46](#)
- filled\_partitions
  - Utility Functions, [13](#)
- free\_entries
  - Ledger Memory Functions, [21](#)
- free\_for\_retotal
  - Ledger Memory Functions, [21](#)
- free\_ledger
  - Ledger Memory Functions, [21](#)
- get\_entries\_from\_filename
  - Ledger Input Functions, [33](#)
- get\_entries\_from\_stream
  - Ledger Input Functions, [33](#)
- get\_entries\_from\_string
  - Ledger Input Functions, [33](#)
- get\_ledger
  - Ledger Input Functions, [34](#)
- get\_names
  - Ledger Memory Functions, [22](#)
- get\_totals
  - Ledger Memory Functions, [22](#)
- I\_CLEARED
  - Totals Macros, [7](#)
- I\_NOT\_THERE\_YET

- Totals Macros, 7
- I\_OVERALL\_BAL
  - Totals Macros, 7
- I\_PENDING
  - Totals Macros, 7
- I\_PENDING\_BAL
  - Totals Macros, 8
- index\_t
  - Return Types, 11
- input\_file
  - Utility Functions, 14
- insert\_filled\_rows
  - Ledger Modify Functions, 26
- insert\_rows
  - Ledger Modify Functions, 26
- Internal Macros, 10
  - ENTRYSIZE, 10
  - EPS, 10
  - FILENAME\_SIZE, 10
  - LINESIZE, 10
  - NFIELDS, 10
- LFAILURE
  - Return Value Macros, 9
- LINESIZE
  - Internal Macros, 10
- LNO
  - Return Value Macros, 9
- LOCKED
  - Status Macros, 42
- LSUCCESS
  - Return Value Macros, 9
- LYES
  - Return Value Macros, 9
- Ledger, 45
  - bank\_totals, 45
  - banks, 45
  - credit\_totals, 45
  - credits, 46
  - entries, 46
  - filename, 46
  - nbanks, 46
  - ncredits, 46
  - npartitions, 46
  - nrows, 46
  - partition\_totals, 46
  - partitions, 46
- Ledger Input Functions, 33
  - get\_entries\_from\_filename, 33
  - get\_entries\_from\_stream, 33
  - get\_entries\_from\_string, 33
  - get\_ledger, 34
  - parse\_char, 34
- Ledger Memory Functions, 20
  - alloc\_entries, 20
  - alloc\_totals, 20
  - copy\_ledger, 21
  - free\_entries, 21
  - free\_for\_retotal, 21
  - free\_ledger, 21
  - get\_names, 22
  - get\_totals, 22
  - new\_ledger, 22
- Ledger Modify Functions, 23
  - clean, 24
  - condense, 24
  - copy\_rows, 24
  - cut\_rows, 25
  - edit\_entry, 25
  - edit\_entry\_noretal, 25
  - edit\_row, 26
  - insert\_filled\_rows, 26
  - insert\_rows, 26
  - map\_to\_coords, 26
  - map\_to\_coords\_colmajor, 28
  - map\_to\_coords\_rowmajor, 28
  - move\_rows, 28
  - paste\_rows, 29
  - permute\_rows, 29
  - remove\_rows, 29
  - rename\_bank, 29
  - rename\_credit, 30
  - rename\_partition, 30
  - repartition, 30
  - retotal, 31
  - sort\_by\_status, 31
  - strip\_ledger, 31
  - swap\_rows, 31
  - trim\_ledger, 32
  - unlock, 32
- Ledger Output Functions, 35
  - print\_ledger\_to\_filename, 35
  - print\_ledger\_to\_stream, 35
  - print\_ledger\_to\_string, 35
  - print\_ledger\_verbose, 36
- Ledger Summary Functions, 37
  - print\_summary\_to\_filename, 37
  - print\_summary\_to\_stream, 37
  - print\_summary\_to\_string, 37
- legal\_amounts
  - Utility Functions, 14
- legal\_double
  - Utility Functions, 14
- legal\_status\_code
  - Utility Functions, 14
- legal\_status\_codes
  - Utility Functions, 15
- locked
  - Utility Functions, 15
- main
  - main.c, 69
- main.c
  - main, 69
- map\_to\_coords
  - Ledger Modify Functions, 26
- map\_to\_coords\_colmajor
  - Ledger Modify Functions, 28

- map\_to\_coords\_rowmajor
  - Ledger Modify Functions, [28](#)
- move\_rows
  - Ledger Modify Functions, [28](#)
- N\_TOTALS
  - Totals Macros, [8](#)
- NEGATIVE\_COLOR
  - Printing Macros, [44](#)
- NFIELDS
  - Internal Macros, [10](#)
- NO\_INDEX
  - Return Value Macros, [9](#)
- NORMAL\_COLOR
  - Printing Macros, [44](#)
- NOT\_THERE\_YET
  - Status Macros, [42](#)
- nbanks
  - Ledger, [46](#)
- ncredits
  - Ledger, [46](#)
- new\_ledger
  - Ledger Memory Functions, [22](#)
- npartitions
  - Ledger, [46](#)
- nrows
  - Ledger, [46](#)
- output\_file
  - Utility Functions, [15](#)
- PARTITION
  - Column Indices, [40](#)
- PENDING
  - Status Macros, [42](#)
- POSITIVE\_COLOR
  - Printing Macros, [44](#)
- PRINT\_EMPTY\_ACCOUNTS
  - Printing Macros, [44](#)
- parse\_char
  - Ledger Input Functions, [34](#)
- partition\_totals
  - Ledger, [46](#)
- partitions
  - Ledger, [46](#)
- paste\_rows
  - Ledger Modify Functions, [29](#)
- permute\_rows
  - Ledger Modify Functions, [29](#)
- print\_ledger\_to\_filename
  - Ledger Output Functions, [35](#)
- print\_ledger\_to\_stream
  - Ledger Output Functions, [35](#)
- print\_ledger\_to\_string
  - Ledger Output Functions, [35](#)
- print\_ledger\_verbose
  - Ledger Output Functions, [36](#)
- print\_summary\_to\_filename
  - Ledger Summary Functions, [37](#)
- print\_summary\_to\_stream
  - Ledger Summary Functions, [37](#)
- print\_summary\_to\_string
  - Ledger Summary Functions, [37](#)
- Printing Macros, [44](#)
  - NEGATIVE\_COLOR, [44](#)
  - NORMAL\_COLOR, [44](#)
  - POSITIVE\_COLOR, [44](#)
  - PRINT\_EMPTY\_ACCOUNTS, [44](#)
  - USE\_COLOR, [44](#)
  - ZERO\_COLOR, [44](#)
- qcmp
  - Utility Functions, [15](#)
- REMOVE
  - Status Macros, [43](#)
- ROW\_SEPARATORS
  - Separator Macros, [41](#)
- remove\_rows
  - Ledger Modify Functions, [29](#)
- rename\_bank
  - Ledger Modify Functions, [29](#)
- rename\_credit
  - Ledger Modify Functions, [30](#)
- rename\_partition
  - Ledger Modify Functions, [30](#)
- repartition
  - Ledger Modify Functions, [30](#)
- retotal
  - Ledger Modify Functions, [31](#)
- Return Types, [11](#)
  - bool\_t, [11](#)
  - color\_t, [11](#)
  - err\_t, [11](#)
  - index\_t, [11](#)
- Return Value Macros, [9](#)
  - LFAILURE, [9](#)
  - LNO, [9](#)
  - LSUCCESS, [9](#)
  - LYES, [9](#)
  - NO\_INDEX, [9](#)
- row\_delim\_char
  - Utility Functions, [16](#)
- row\_delim\_str
  - Utility Functions, [16](#)
- STATUS
  - Column Indices, [40](#)
- Separator Macros, [41](#)
  - COLUMN\_SEPARATORS, [41](#)
  - ROW\_SEPARATORS, [41](#)
- small\_norm
  - Utility Functions, [16](#)
- sort\_by\_status
  - Ledger Modify Functions, [31](#)
- space
  - Utility Functions, [16](#)
- src/alloc\_entries.c, [47](#)

- src/alloc\_totals.c, [47](#)
- src/clean.c, [48](#)
- src/col\_delim\_char.c, [49](#)
- src/col\_delim\_str.c, [49](#)
- src/color.c, [50](#)
- src/condense.c, [50](#)
- src/copy\_ledger.c, [51](#)
- src/copy\_rows.c, [51](#)
- src/cut\_rows.c, [52](#)
- src/edit\_entry.c, [53](#)
- src/edit\_entry\_noretal.c, [53](#)
- src/edit\_row.c, [54](#)
- src/filled\_partitions.c, [54](#)
- src/free\_entries.c, [55](#)
- src/free\_for\_retotal.c, [55](#)
- src/free\_ledger.c, [56](#)
- src/get\_entries\_from\_filename.c, [57](#)
- src/get\_entries\_from\_stream.c, [57](#)
- src/get\_entries\_from\_string.c, [58](#)
- src/get\_ledger.c, [58](#)
- src/get\_names.c, [59](#)
- src/get\_totals.c, [59](#)
- src/input\_file.c, [60](#)
- src/insert\_filled\_rows.c, [61](#)
- src/insert\_rows.c, [61](#)
- src/ledger.h, [62](#)
- src/legal\_amounts.c, [66](#)
- src/legal\_double.c, [67](#)
- src/legal\_status\_code.c, [67](#)
- src/legal\_status\_codes.c, [68](#)
- src/locked.c, [68](#)
- src/main.c, [69](#)
- src/map\_to\_coords.c, [70](#)
- src/map\_to\_coords\_colmajor.c, [70](#)
- src/map\_to\_coords\_rowmajor.c, [71](#)
- src/move\_rows.c, [71](#)
- src/new\_ledger.c, [72](#)
- src/output\_file.c, [72](#)
- src/parse\_char.c, [73](#)
- src/paste\_rows.c, [74](#)
- src/permute\_rows.c, [74](#)
- src/print\_ledger\_to\_filename.c, [75](#)
- src/print\_ledger\_to\_stream.c, [75](#)
- src/print\_ledger\_to\_string.c, [76](#)
- src/print\_ledger\_verbose.c, [76](#)
- src/print\_summary\_to\_filename.c, [77](#)
- src/print\_summary\_to\_stream.c, [78](#)
- src/print\_summary\_to\_string.c, [78](#)
- src/qcmp.c, [79](#)
- src/remove\_rows.c, [79](#)
- src/rename\_bank.c, [80](#)
- src/rename\_credit.c, [80](#)
- src/rename\_partition.c, [81](#)
- src/repartition.c, [82](#)
- src/retotal.c, [82](#)
- src/row\_delim\_char.c, [83](#)
- src/row\_delim\_str.c, [83](#)
- src/small\_norm.c, [84](#)
- src/sort\_by\_status.c, [84](#)
- src/space.c, [85](#)
- src/standalone.c, [86](#)
- src/str\_equal.c, [86](#)
- src/str\_strip.c, [87](#)
- src/strip\_ledger.c, [87](#)
- src/swap\_rows.c, [88](#)
- src/trim\_ledger.c, [88](#)
- src/unique.c, [89](#)
- src/unlock.c, [90](#)
- src/untotaled.c, [90](#)
- src/usage.c, [91](#)
- src/user\_settings.h, [91](#)
- src/which.c, [93](#)
- src/which\_bank\_total.c, [93](#)
- src/which\_credit\_total.c, [94](#)
- standalone
  - Top Level Functions, [39](#)
- Status Macros, [42](#)
  - CREDIT\_CHARGED, [42](#)
  - CREDIT\_PENDING, [42](#)
  - LOCKED, [42](#)
  - NOT\_THERE\_YET, [42](#)
  - PENDING, [42](#)
  - REMOVE, [43](#)
- str\_equal
  - Utility Functions, [17](#)
- str\_strip
  - Utility Functions, [17](#)
- strip\_ledger
  - Ledger Modify Functions, [31](#)
- swap\_rows
  - Ledger Modify Functions, [31](#)
- Top Level Functions, [39](#)
  - standalone, [39](#)
- Totals Macros, [7](#)
  - I\_CLEARED, [7](#)
  - I\_NOT\_THERE\_YET, [7](#)
  - I\_OVERALL\_BAL, [7](#)
  - I\_PENDING, [7](#)
  - I\_PENDING\_BAL, [8](#)
  - N\_TOTALS, [8](#)
- trim\_ledger
  - Ledger Modify Functions, [32](#)
- USE\_COLOR
  - Printing Macros, [44](#)
- unique
  - Utility Functions, [17](#)
- unlock
  - Ledger Modify Functions, [32](#)
- untotaled
  - Utility Functions, [17](#)
- usage
  - Utility Functions, [18](#)
- Utility Functions, [12](#)
  - col\_delim\_char, [13](#)
  - col\_delim\_str, [13](#)

- color, [13](#)
- filled\_partitions, [13](#)
- input\_file, [14](#)
- legal\_amounts, [14](#)
- legal\_double, [14](#)
- legal\_status\_code, [14](#)
- legal\_status\_codes, [15](#)
- locked, [15](#)
- output\_file, [15](#)
- qcmp, [15](#)
- row\_delim\_char, [16](#)
- row\_delim\_str, [16](#)
- small\_norm, [16](#)
- space, [16](#)
- str\_equal, [17](#)
- str\_strip, [17](#)
- unique, [17](#)
- untotaled, [17](#)
- usage, [18](#)
- which, [18](#)
- which\_bank\_total, [18](#)
- which\_credit\_total, [18](#)

which

- Utility Functions, [18](#)

which\_bank\_total

- Utility Functions, [18](#)

which\_credit\_total

- Utility Functions, [18](#)

ZERO\_COLOR

- Printing Macros, [44](#)