Beancount Example & Tutorial

[Martin Blais](mailto:blais@furius.ca), October 2014

<http://furius.ca/beancount/doc/example>

[Example File Generator](#_pqvy7j6aqf55)

[Converting to Ledger Input](#_t6f7zhkj0d1k)

[Profile of Example User](#_hd75sz85rbpr)

[Future Additions](#_y4lz0y3lslyj)

[Tutorial](#_odqvifnh2doi)

[Generate an Example File](#_3490lk5w1n6v)

[Generating Reports](#_are2unbhsak6)

[Generating Balances](#_cumoa9w4g56o)

[Generating a Balance Sheet and Income Statement](#_iot9y9kvcep1)

[Journals](#_b5oljnqxs5zj)

[Holdings](#_6k9snjuobfd7)

[Other Reports](#_5vkugtxpso60)

[Other Formats](#_d920tix560mx)

[Viewing Reports through the Web Interface](#_inkodk2pjywa)

[The Future of Beancount Reports](#_vlh1pclqsgq8)

# Example File Generator

Beancount has a command to generate a few years of a realistic user’s historical entries:

**bean-example** > example.beancount

The script checks if the output fails to process cleanly in Beancount (if this occurs, try rerunning or contact the author at [blais@furius.ca](mailto:blais@furius.ca)). Note that there is an option to fix the random generator’s seed to be used to generate the entries.

You can generate multiple years of ledger data for this user; see available options with

bean-example **--help**

The purpose of the script is:

* To provide a realistic corpus of anonymous data for users to experiment with, to kick the tires of Beancount, generate reports, etc.
* To compare reports with those generated by Ledger.
* As an example file to be used in this tutorial.
* As input for stress testing Beancount.

## Converting to Ledger Input

It should be possible to convert example files generated from the script to Ledger syntax, like this:

**bean-report** example.beancount **ledger** > example.lgr

If you encounter any problems with the conversion, please let me know. While I have an automated test to check that the conversion succeeds, I’m focusing my efforts on Beancount itself and I’m not using Ledger much other than for the occasional comparison or for illustrating something on the mailing-list.

# Profile of Example User

Here’s a high-level profile of this user. In the text that follows, I’ll use the masculine “he” for simplicity’s sake.

* He lives somewhere in the USA and uses a single operating currency: “USD”. He rents an apartment.
* **Income**: He is a software engineer (sorry I could not resist), receives a modest salary income from a software company.
  + He is paid bi-weekly.
  + He maximizes contributions to his company’s 401k plan. His company offers a match to his contributions.
  + His salary is paid as a direct deposit to his checking account.
  + He benefits and pays premiums for a health insurance plan provided by his employer.
  + He tracks his vacation hours explicitly.
* **Expenses**:
  + **Checking account**: He pays his apartment’s rent by check. He also pays his electricity, internet ISP and mobile phone bills directly from his checking account. His bank also takes a monthly bank fee.
  + **Credit cards**: He also has other regular expenses, such as groceries and restaurant outings with friends, which he pays from his credit card.
* **Assets**:
  + **Banking**: He has an account at a major US bank.
  + **Retirement account**: His 401k account is held at Fidelity or Vanguard and is allocated in its entirety to two funds, a mix of stocks and bonds. There are never any sales in this account. This account is held at average cost.
  + **Other investments**: Extra money that he is able to save is transferred from his checking account towards a taxable investment account in which he purchases a mix of stocks and ETFs, and occasionally makes sales and realizes profits.
* **Liabilities**:
  + **Credit cards**: Hehas one or two credit cards issued by another bank.
* **Events/Tags**: He takes a short vacation or two per year, where he travels somewhere fun.

## Future Additions

Note that as I beef up the example generator script, this profile will increase in complexity. If there is a particular situation from the [cookbook](http://furius.ca/beancount/doc/cookbook) or otherwise that you’d like to see in the example file, please let me know (a patch is the best way to let me know).

* Generate fake PDF files for documents (from an option) along with suitable document directives for them, most of them implicit but some explicit.
* Generate more expenses:
  + Some amount of realistic cash expenses.
  + Add a vehicle: car repairs, gas, etc. Remove tram example, it is too urban-specific.
  + Add donations for charity.
  + Add a second credit card and vary sources of expenses.
* Add some real uses for links, for the 401k match, for instance, the employer contributions should be linked with the employee contributions. Or other ones.
* Generate a loan and payments for a student loan. Having a large liability is a common case, I’d like to represent that, even if it’s not a mortgage.
* Convert retirement accounts to book using the average cost method, and add corresponding fees which would otherwise be impossible to track with explicit lots.
* Add tracking of health care expenses as per the cookbook.
* Add foreign accounts in another currency and transactions and transfers in that currency.

# Tutorial

This section provides basic examples of generating reports with Beancount, on the typical example file that would be output by it. The list of reports here is not meant to be exhaustive.

You may follow the tutorial and generate your own example file, or look at the files from [beancount/examples/tutorial](http://bitbucket.org/blais/beancount/src/tip/examples/tutorial/) if you prefer.

## Generate an Example File

Begin by generating the example file:

bean-example > **example.beancount**

Open example.beancount and examine it.

Next, before we begin generating reports, verify that the file loads without any errors:

**bean-check** example.beancount

## It should return quietly, without outputting anything (bean-check only writes errors when there are some, otherwise on success it writes nothing).

## Generating Reports

Let’s generate a report of the final balances of all accounts [[output]](http://furius.ca/beancount/examples/tutorial/balances.output):

**bean-report** example.beancount **balances**

As you can see, the bean-report script has subcommands for the various reports it generates. To list the available reports, use --help-reports [[output]](http://furius.ca/beancount/examples/tutorial/help-reports.output):

bean-report **--help-reports**

To list the options available for a particular report, use --help on it [[output]](http://furius.ca/beancount/examples/tutorial/help-subcmd.output):

bean-report example.beancount balances **--help**

The report-specific options vary for each report type.

You can also list the global options for the script [[output]](http://furius.ca/beancount/examples/tutorial/help-global.output):

bean-report **--help**

The global options allow you to specify the desired output format for the reports. To see which formats are available for which report, use --help-formats [[output]](http://furius.ca/beancount/examples/tutorial/help-formats.output):

bean-report **--help-formats**

## Generating Balances

Good, so we know how to generate a report of balances for all accounts. This is a pretty detailed list of accounts though. Let’s just restrict the output to the accounts that we’re interested in [[output]](http://furius.ca/beancount/examples/tutorial/balances-restrict.output):

bean-report example.beancount balances **-e ETrade**

Here you can view the number of units held in each of the investment subaccounts. To render the cost, use the --cost option [[output]](http://furius.ca/beancount/examples/tutorial/balances-restrict-cost.output):

bean-report example.beancount balances -e ETrade **--cost**

## Formatting Tools

Sometimes it’s nice to render a hierarchical list of accounts as a tree. You can use the “treeify” tool provided by Beancount to do this:

bean-report example.beancount balances **| treeify**

This tool will work on any column of data that looks like a column of account names (you can also configure it work with filenames as well, or other patterns).

## Generating a Balance Sheet and Income Statement

Let us generate a balance sheet:

bean-report example.beancount **balsheet**

Unfortunately, the only output format supported for it at this point is HTML. Also, filtering for balance sheet from the command-line is not supported. Generate this to a file and open a browser to it [[output]](http://furius.ca/beancount/examples/tutorial/balsheet.output):

bean-report example.beancount balsheet **> /tmp/balsheet.html**

You can to the same for income statements:

bean-report example.beancount **income** **> /tmp/income.html**

## Journals

You can also generate journals (in Ledger parlance, these are “registers”). Let’s look at a checking account postings, for instance [[output]](http://furius.ca/beancount/examples/tutorial/journal.output):

bean-report example.beancount **journal -a Assets:US:BofA:Checking**

To render a column of running balances, add the --balance option [[output]](http://furius.ca/beancount/examples/tutorial/journal-with-balance.output):

bean-report example.beancount journal -a Assets:US:BofA:Checking **--balance**

The inventories are rendered in the number of units they’re holding [[output]](http://furius.ca/beancount/examples/tutorial/invest.output):

bean-report example.beancount journal -a **Assets:US:ETrade:GLD**

If you want to render the values at cost, use the --cost option [[output]](http://furius.ca/beancount/examples/tutorial/invest-with-cost.output):

bean-report example.beancount journal -a Assets:US:ETrade:GLD **--cost**

To render journals, you will typically restrict the set of postings that you want to view. If you did not, the postings would render no change, because all the legs of the transactions would be applied to the running balance and the total would be zero. But try it, nonetheless [[output]](http://furius.ca/beancount/examples/tutorial/journal-unrestricted.output):

bean-report example.beancount **journal --balance**

## Holdings

There are a variety of ways to obtain aggregations for the total list of holdings. List the detailed holdings [[output]](http://furius.ca/beancount/examples/tutorial/holdings.output):

bean-report example.beancount **holdings**

To aggregate the holdings by account, do this [[output]](http://furius.ca/beancount/examples/tutorial/holdings-by-account.output):

bean-report example.beancount holdings **--by=account**

Because it’s common practice to create and use a dedicated Beancount subaccount name for each commodity held in a real-world account, we can refine this further to aggregate to the parent (“root”) accounts of those subaccounts [[output]](http://furius.ca/beancount/examples/tutorial/holdings-by-root-account.output):

bean-report example.beancount holdings **--by=root-account**

We can aggregate by commodity [[output]](http://furius.ca/beancount/examples/tutorial/holdings-by-commodity.output):

bean-report example.beancount holdings **--by=commodity**

Or by the currency in which the commodities are quoted, which gives us a glimpse into our currency exposure [[output]](http://furius.ca/beancount/examples/tutorial/holdings-by-currency.output):

bean-report example.beancount holdings **--by=currency**

Finally, we can aggregate all the holdings to obtain net worth [[output]](http://furius.ca/beancount/examples/tutorial/networth.output):

bean-report example.beancount **networth**

There are a few more options for converting amounts to a common currency. See help for details.

## Other Reports

There are many other miscellaneous reports available. Try a few of those.

Listing all accounts [[output]](http://furius.ca/beancount/examples/tutorial/accounts.output):

bean-report example.beancount **accounts**

Listing events [[output]](http://furius.ca/beancount/examples/tutorial/events.output):

bean-report example.beancount **events**

Number of directives by type [[output]](http://furius.ca/beancount/examples/tutorial/stats-directives.output):

bean-report example.beancount **stats-directives**

Number of postings by type [[output]](http://furius.ca/beancount/examples/tutorial/stats-postings.output):

bean-report example.beancount **stats-postings**

## Other Formats

Reports may support various output formats. You can change the output format with the --format (-f) global option [[output]](http://furius.ca/beancount/examples/tutorial/holdings-csv.output):

bean-report **-f csv** example.beancount holdings

Note that “beancount” and “ledger” are valid output formats: they refer to the Beancount and Ledger input syntaxes.

## Viewing Reports through the Web Interface

The original way to access reports in Beancount is via its web interface, that serves to a local web server on your machine. Serve the example file like this:

**bean-web** example.beancount

Then navigate with a web browser to <http://localhost:8080>. From there, you can click on any number of filtered views and access some of the reports previously demonstrated. For example, click on a year view; that will provide balance sheets and income statements and various other reports for this subset of transactions.

The bean-web tool has many options for restricting what is being served. (Note that by default the server listens only for connections from your computer; if you want to host this on a web server and accept connections from anywhere, use --public).

*Note: There exists a separate project which provides a better web interface than the one which comes with Beancount:* [*Fava*](https://github.com/aumayr/fava)*. You might want to check it out.*

# The Future of Beancount Reports

I find my command-line reports above to be rather unsatisfying, from the point-of-view of customizability. They bother me a lot. This is largely because I’ve been happy and satisfied with the capabilities of the web interface to Beancount. As I’m beginning to use the command-line interface more and more, I’m finding myself desiring for more explicit and well-defined way to apply all the filters I have available from the web interface, and more. For this reason, I’m [currently implementing a query language that looks similar to SQL](http://furius.ca/beancount/doc/proposal-filtering). This syntax will allow me to remove all the custom reports above and to replace them by a single consistent query syntax. This work is underway.

All the holdings reports and even their internal objects will be replaced by the SQL-like query syntax as well. Holdings need not be treated separately: they are simply the list of positions available at a particular point in time, and along with suitable accessors from the query syntax and support for aggregations over their many columns, we should be able to generate all of those reports equivalently and simplify the code.

*— Martin Blais, October 2014*