Investors' report

David Kastrup

July 2013

Somewhat embarrassingly, the report for July is finally done. Again, the first two entries are from distributing one-time payments across several months, and the first of those will keep running for well into next year (if not longer). Once it runs out, it will be time to reevaluate where I stand, but for now it takes away considerable pressure as the level from regular small payments is not really on the increase.

Without further ado, here are the sums:

July

Fixed payments (€)
300
250
2×200
100
50
2×30
4×25
2×20 5×10
1350

Variable plans (€)
25 + 0×50 (target €1200 exceeded in June)

Totals €1375

Since the August report is still to come, I can at least mention the totals for August: looks like €1121 at first glance. Which is consistent with the July results if we take into account that the July numbers contained the last €250 from a one-time May payment. It's not as bad as I expected given that I am still running one report late, but then it's not exactly fabulous either. One regular contributor for over a year has received a large financial hit right now and has apologized that he is forced to discontinue his support. Of course, I am glad for the time he has been able to help, contributing to a sizable piece of keeping me going. It should not really be a reason to apologize.

Which reminds me to say a big "Thank you!" to all the other faithful of this kind who contribute like clockwork. Whenever I find myself having dropped the ball regarding my

reports and/or the work I've managed to put in and am actually more or less afraid of daring to look at my account, I regularly am gratefully surprised that there are people more dependable than I feel myself to be.

Development results

Developments in July, listed with

```
git shortlog origin --since 2013/07/01 --until 2013/08/01 -n
```

in a checkout of the project repository list 20 commits by me, which is somewhat lacklustre. As explained in the last report, this is mostly due to myself trying to focus on algorithmic detail of the skyline merging. Somewhat sad to say, no commits have yet resulted from this effort: while I have a plan for the algorithm worked out, I got side-tracked so much when doing the actual implementation that I finally turned to more 'productive' endeavors. That's sort of a common trap when trying to create 'over-general' algorithmic skeletons/templates in C++ that could theoretically be reused for other purposes but in practice aren't. I need to refocus on doing this in a 'non-general' manner. In the meantime, this is still pending completion. It's actually pending most of the actual coding since the generalized framework from which to hang the code is not coherently finished.

Turning from that partly self-inflicted, partly language-promoted programmer's block, it's worth taking a look at the actually committed unrelated work. The long unfinished code for typesetting accordion registers has been wrapped up and committed, there were some small parser changes, and some changes in internal data structures apart from several bug fixes. There was also work to let our documentation continue to be compiled with newer Ghostscript versions: the actual fix was quite small, but it took a lot of debugging and planning and experimentation to get to a somewhat crude but operative fix.

Ongoing tasks

While it may seem like LilyPond 2.18 is sort of a running gag in recent reports, we are making a lot of headway towards getting the remaining issues that are unfit for making a stable release under wraps. So I'm pretty confident that we can cut the "Stable Release Branch" reasonably soon and get to a stable release before the money runs out.

Joking aside, this will mark the first stable release for which my work has been completely financed by the LilyPond community.

Perspectives

The developer meeting in Waltrop that I mentioned in the last report had first been postponed for a month, then cancelled completely due to a dearth of participants. On the first scheduled date, I had an informal gettogether with two music lovers using LilyPond as a mission-critical part of for their own projects. Characterizing those in a

nutshell, they were the Scora project for synchronized music stands, and the Philomelos project for colloborative web-based management of scores.

It was interesting to see and discuss where the projects are headed and what LilyPond can (and likely should) do to make it easier for them, where it excels and where it can be somewhat of a roadblock.

It's also somewhat sobering to realize that if someone were to say "Here are a million Euros, make LilyPond go somewhere" that LilyPond's development and program structure would make it rather challenging to find areas of work that are non-specialized enough that one can throw programming talent available on the free market at it.

Probably a batch MusicXML to ly and back infrastructure could be done in a stock manner, and one would likely want to revive ailing web infrastructure, the translation project, get the infrastructure for moving and maintaining Mutopia operative on newer versions of LilyPond, set up colloborative editing projects, work on input tools including Midi and so forth and so on. There is actually a lot of stuff that can be done *surrounding* LilyPond. It's just that the *core* is rather hard to refactor and move forward using a 'standard' talent pool.

Thanks

as always for making it possible for me to continue my work, and hopefully for making me help with getting to a more timely release of LilyPond 2.18. Version 2.16 took about two years to complete, and it definitely looks like we'll be quite faster this time around.

Thanks for all of you for enabling my work on LilyPond!

David Kastrup