TeXworks — As you like it

Stefan Löffler

Abstract

TeXworks is an ongoing project to create a simple yet flexible editor that "lowers the entry barrier to the TeX world". This article introduces the new TeXworks 0.4 series and discusses several key advancements, in particular the new scripting support.

1 Introduction

The TeX world is complex. "Beautifully complex", the expert will say. "Dreadfully complex", would be a newcomer's more likely choice of words. This is where TeXworks comes in. Its motto: "*to lower the entry barrier to the TeX world*".

The TeXworks project was launched in 2007, out of discussions between Jonathan Kew, Karl Berry, Dick Koch, and others at several TUG meetings. The plan was to create a new editor, modeled along the lines of the award-winning TeXShop application for Mac OS X, but as an open-source, cross-platform program. TUG sponsored some of the initial development.

After two years of development, the first stable release of TeXworks, 0.2.0, was published in 2009. From that point on, odd minor version numbers (0.1, 0.3, ...) indicated unstable development code, while even numbers (0.2, 0.4, ...) indicated stable releases for general use.

The hope from there was to speed up development and produce stable releases more often. Due to other responsibilities, Jonathan had less and less time for coding, however, and the next stable release had to be postponed. In 2010, Jonathan handed over a large part of the development responsibilities to me, a fairly regular contributor of patches for quite a while. After wrapping up the dangling threads, the first of the stable 0.4 releases was published in the first half of 2011. Apart from many bug fixes and enhancements, one theme dominated this series: scripting.

2 Scripting TeXworks

All the world's a stage
And all the men and women merely players;
They have their exits and their entrances,
And one man in his time plays many parts.
— William Shakespeare, *As You Like It*

For this project, this can be interpreted as: TeXworks is only the core program, the basis upon which every user can personalize the TeX editor to their liking and tailor it to meet individual needs.

With the growing popularity of TeXworks and the wide nature of the TeX user base, more and more requests for specific features started coming in. Either for a generally useful addition, to meet an expert's specific needs, to use TeXworks in some completely unforeseen way, or sometimes for things relevant only to very few, limited, special situations.

In short, with the growing user base, so came an increasing number of divergent ideas for the project's growth, and it was clear that not everybody's wishes could be accommodated. We realized that if we adopted ideas that were too specialized, or introduced too great a complexity into the user interface, we would fail in the primary purpose of providing a straightforward editor — one which would not scare off new users. Thus, we faced a dilemma concerning extending the usefulness and versatility of the project, while somehow keeping the standard interface "clean".

The only flexible solution seemed to be to allow the actual users to change TeXworks to their own liking and needs. Since not everyone is proficient in C++ programming, and a large amount of forked code would be impossible to maintain, letting users *script* new features that were not part of the core application was identified as the best answer, and a lot of effort has gone into that. Scripts can be added any time as they do not need to be compiled. They are simple text files outside the main code, can be deployed as needed, and work on all platforms.

Early attempts at a scripting engine were rudimentary at best. As a proof of concept, it was possible to insert and modify some text from a script, but the C++ internals were very clumsy and hard to maintain.

The real breakthrough came through the discovery that Qt — the Nokia programming framework on which TeXworks is based — allows dynamic access to almost all parts of the core program. There were some coding tricks involved, but exploiting this existing mechanism saved us the work of creating wrappers for each function and variable that script writers may access.

The Qt framework even allows script writers to create forms and dialog windows to interact with the user directly. Nokia provides a free tool, Qt Creator, an IDE with widgets and components, to create these additional user interfaces.

The second major advance was the restructuring of the scripting code to accommodate plugins for additional scripting languages. Presently, apart from the built-in, JavaScript-like QtScript, Lua and Python are available via plugins (on operating systems that support this mechanism). This approach also enables programmers to easily add additional scripting languages to TeXworks.

3 How scripts work

The easiest way to use scripts to adapt TeXworks to your liking is to get a ready-made script and simply drop it into the TeXworks 'scripts' folder. This can easily be found using the "Show Scripts Folder" menu item. After dropping your script files in, click "Reload Script List" and you're ready to go.

TeXworks scripts come in two varieties: "standalone" scripts and "hook" scripts. "Standalone" scripts appear as menu items in the "Scripts" menu (or one of its submenus). Running such a script is done by clicking on the menu item, or by using a shortcut key (sequence), if one is assigned.

"Hook" scripts, on the other hand, are not directly invoked by the user. Instead, TeXworks runs them automatically in certain situations. For example, a hook script could run automatically after a typeset process completes, parse the log output, and present errors or warnings in a user-friendly way.

Scripts can also access files on your hard disk and even execute system commands. To help spare you any severe security problems should an untrusted script be inadvertently run, these features are disabled by default. This can prevent some (advanced) scripts from working properly, however. To enable these advanced features, a one-time authorization must be made in the TeXworks preferences dialog.

Other, existing script libraries can be modified and used (e.g., phpjs [4]). Among many other things, script system commands can be (and have been) fashioned to: retrieve information from databases or bibliography citations, interact with utilities like ImageMagick, and provide additional visual help for LaTeX and others. Combined with script-writer-designed dialogs and forms, the possibilities are very wide, and no C++ knowledge is required!

If you're interested in writing scripts of your own to make your or your colleague's life that little bit easier — whether to insert the same text over and over again by a simple key sequence, or write complex scripts for providing input-driven templates — there are a number of resources out there to learn scripting. Of course, knowing your way around one of the supported languages (JavaScript, Lua, Python) in general helps. Other than that, have a look at the TeXworks manual [5] for a general, more in-depth discussion of how to use scripts, and Paul A. Norman's excellent overview of how to manipulate TeXworks from within a script [6].

4 Other news about TeXworks

The 0.4 stable series has brought in many other improvements as well. Those who have used previous releases may notice that quite a lot of effort has been put into further enhancing usability. For one, the presentation of spell checking languages has been improved significantly. Now, human readable names are shown instead of ISO language codes, and languages no longer show up multiple times. In addition, a "follow focus" option has been implemented that can keep the cursor position in the editor and the preview windows synchronized. Syntax highlighting has also been enhanced — it is now possible to set some font modifiers (bold, italic, etc.) and background colors.

TeXworks's core has also seen some improvements. Most notably, a new command line parser and an automatic updating mechanism for TeXworks resource files have been added. The command line parser enables better integration with other tools that call TeXworks (e.g., other editors, previewers, or the operating system).

The automatic updating mechanism will allow future versions of TeXworks to upgrade resources like auto-completion files, or syntax highlighting definitions (provided the user does not intervene, of course). Previously, it was necessary to find and delete files manually to cause their update.

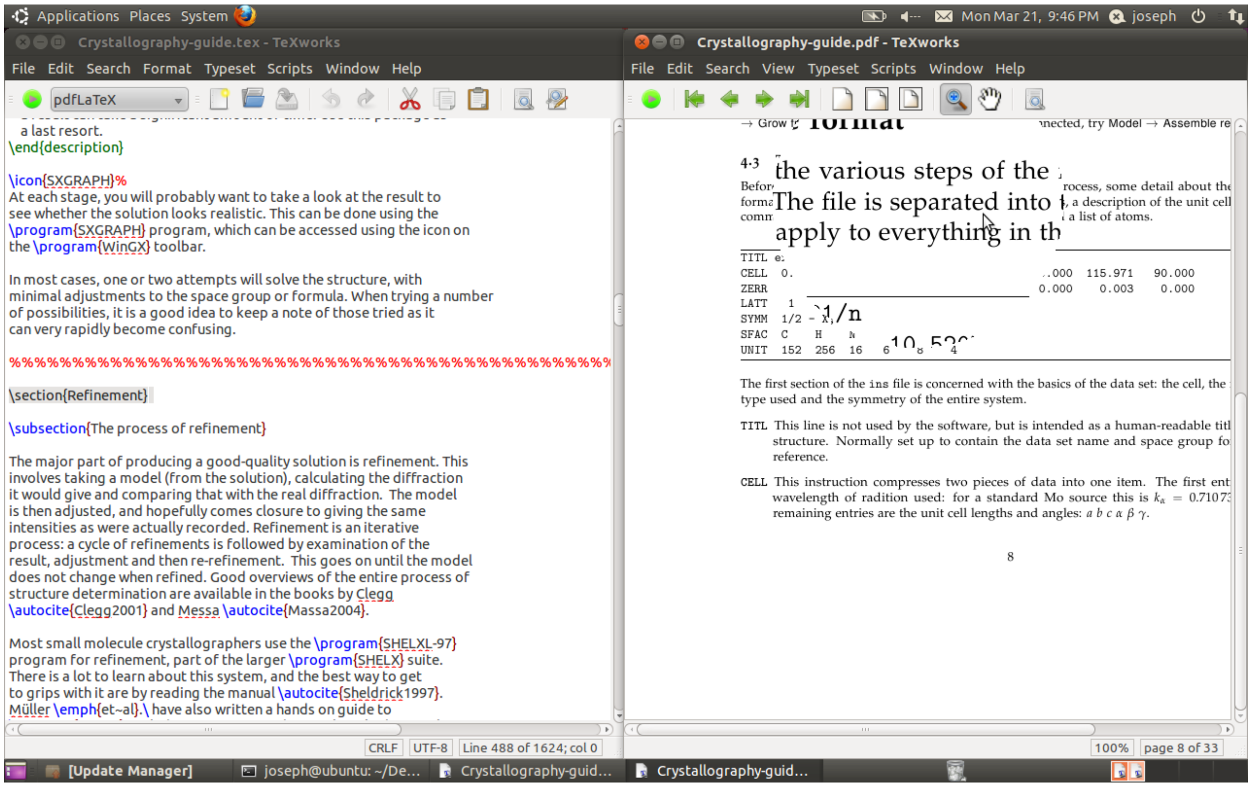
This summary of some of the features that stand out to me is expanded in a more complete overview on TeXworks's home page [2]. The screenshot in figure 1 shows TeXworks in typical usage.

5 Outlook

Far from this being the end of the development of TeXworks itself, or of its scripting support, a number of ideas are being canvassed, such as script bundles that can perform a multitude of (typically related) tasks, or scripts that can run in the background — e.g., to perform or monitor a lengthy task — while the rest of TeXworks can be used normally. In addition, there are plans to allow scripts to modify the user interface — e.g., by supplying toolbar icons or context menu entries — instead of just showing up in a long list of items in the scripts menu.

Beyond the 0.4 series scripting development, other great ideas have been piling up as well: preview window improvements, tabbed editing, code folding, and project management support, to name only a few. Development for the near future will happen in the 0.5 series (which will become the 0.6 stable release eventually), and will likely focus on areas that have been put on hold in the attempt to get the best out of scripting for TeXworks 0.4.

Hopefully, this has got you interested in the TeXworks project (or, if you have already been using it, has been a helpful update). If you want to try TeXworks out for yourself or upgrade from an earlier version, head over to the home page [2] and


Applications Places System
Mon Mar 21, 9:46 PM joseph
Crystallography-guide.tex - TeXworks
File Edit Search Format Typeset Scripts Window Help
pdfLaTeX
Crystallography-guide.pdf - TeXworks
File Edit Search View Typeset Scripts Window Help
[Update Manager]
joseph@ubuntu: ~/De...
Crystallography-guid...
Crystallography-guid...

```
  a last resort.
\end{description}

\icon{SXGRAPH}%
At each stage, you will probably want to take a look at the result to
see whether the solution looks realistic. This can be done using the
\program{SXGRAPH} program, which can be accessed using the icon on
the \program{WinGX} toolbar.

In most cases, one or two attempts will solve the structure, with
minimal adjustments to the space group or formula. When trying a number
of possibilities, it is a good idea to keep a note of those tried as it
can very rapidly become confusing.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

\section{Refinement}

\subsection{The process of refinement}

The major part of producing a good-quality solution is refinement. This
involves taking a model (from the solution), calculating the diffraction
it would give and comparing that with the real diffraction.  The model
is then adjusted, and hopefully comes closure to giving the same
intensities as were actually recorded. Refinement is an iterative
process: a cycle of refinements is followed by examination of the
result, adjustment and then re-refinement.  This goes on until the model
does not change when refined. Good overviews of the entire process of
structure determination are available in the books by Clegg
\autocite{Clegg2001} and Messa \autocite{Massa2004}.

Most small molecule crystallographers use the \program{SHELXL-97}
program for refinement, part of the larger \program{SHELX} suite.
There is a lot to learn about this system, and the best way to get
to grips with it are by reading the manual \autocite{Sheldrick1997}.
Müller \emph{et~al}.\ have also written a hands on guide to
```

CRLF UTF-8 Line 488 of 1624; col 0

→ Grow ℓ format nected, try Model → Assemble re

4.3 the various steps of the
Befor rocess, some detail about the
forma The file is separated into t, a description of the unit cell
comm a list of atoms.
apply to everything in th

```
TITL e:
CELL  0.                                  ..000  115.971   90.000
ZERR                                      0.000    0.003    0.000
LATT   1
SYMM  1/2 - X, 1/n
SFAC  C    H    N
UNIT  152  256  16   6 10 8 52 4
```

The first section of the ins file is concerned with the basics of the data set: the cell, the type used and the symmetry of the entire system.

TITL This line is not used by the software, but is intended as a human-readable titl structure. Normally set up to contain the data set name and space group fo reference.

CELL This instruction compresses two pieces of data into one item. The first ent wavelength of radition used: for a standard Mo source this is k_α = 0.71073 remaining entries are the unit cell lengths and angles: a b c α β γ.

8

100% page 8 of 33

Figure 1: TEXworks source and preview windows on Ubuntu, with area magnified.

grab a copy for your operating system — it's usually quite simple. And if you want to learn more about TEXworks, or perhaps would like to help in its improvement — by giving feedback, translating, writing manuals, contributing code, or any other way — be sure to check out the pointers given in the References section below.

6 Acknowledgements

I want to say a big "thank you" to Paul Norman, who helped in the preparation of this article; to Joseph Wright for providing the screenshot; and to the immensely supportive TEXworks community without which this project wouldn't be where it is today. And of course to Jonathan Kew for initiating the program, maintaining it, and mentoring me.

◇ Stefan Löffler
Döblinger Hauptstraße 13
1190 Wien
Austria
st.loeffler (at) gmail.com

References

[1] TEXworks development home page. http://code.google.com/p/texworks/.

[2] TEXworks home page. http://www.tug.org/texworks/.

[3] TEXworks mailing list. http://lists.tug.org/texworks.

[4] Use PHP functions in JavaScript. http://phpjs.org/.

[5] Alain Delmotte and Stefan Löffler. A short manual for TEXworks. Bundled with TEXworks.

[6] Paul A. Norman. TEXworks scripting. http://twscript.paulanorman.com/docs/index.html.