TLaunch, the TEX Live Launcher for Windows

Siep Kroonenberg

Abstract

The TEX Live Launcher offers Windows users of a network TEX Live installation similar conveniences as a locally-installed TEX Live. It is easy to integrate additional TEX-related software.

This paper describes the launcher and its configuration. As an example, it shows how it is used at the Rijksuniversiteit Groningen.

1 Overview

The TEX Live launcher gives users on Windows workstations easy access to a TEX Live installation already present on the network.

The launcher interface contains menus and buttons to invoke programs, and to access related local and online resources (see figure 1).

It also takes care of the usual Windows-specific configuration: at first run, TEX Live is added to the search path and relevant filetype associations are set up.

Because of prior experience with users running the initializer or installer when they really want to run the already initialized or installed TEX Live, I opted for a launcher that configures itself automatically, without requiring a separate initialization step.

Users can replace the default TEX editor from within the launcher interface, either with an editor defined in an ini file or with a third-party editor present on the filesystem (see figure 2).

For the sake of full access to the Windows API, I wrote the launcher in C. It has no dependencies whatsoever, aside from a TEX Live installation and Windows version 7 or later.

The launcher makes it easy for the TEX Live installation maintainer to add menu- or button controls and filetype associations for additional TEX-related software.

Filetypes, menus and buttons are defined in a Windows ini file. If necessary, pre- and post configuration script files can be configured as well.

The ini file included in TEX Live provides functionality more or less equivalent to the classic Windows TEX Live installation.

In the following sections, we have a more detailed look at the launcher and its configuration. The package documentation contains the full details.

Section 6 looks at the TEX Live installation at the Rijksuniversiteit Groningen, for which the launcher was created.

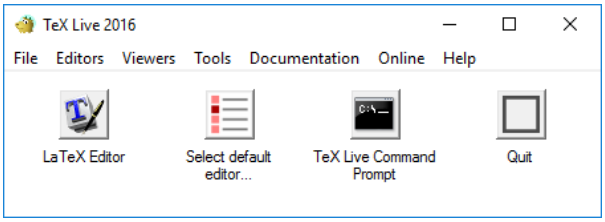

Figure 1: The default TEX Live Launcher

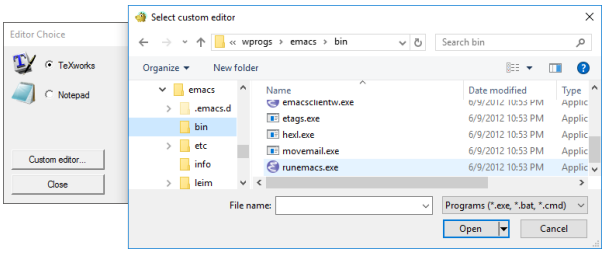

Figure 2: Selecting a custom editor

2 The ini file

The launcher reads its configuration from a conventional Windows ini file with sections, definitions and comment lines. If the TEX Live installation contains Windows binaries, then `tlaunch.exe` will be in the `bin/win32` directory, and `tlaunch.ini` in `texmf-dist/web2c`.[1] A custom ini file, supporting different software or with localized strings, can be placed in a higher-priority tree.

It is also possible to place `tlaunch.exe` and `tlaunch.ini` together in the root of the installation.

In general, entries which refer to non-existent items are silently ignored.

2.1 Strings

There is a Strings section for string variables. String variable names are case-insensitive. Some string variables, such as `%TLCONFIG%`, are required. This variable indicates the directory where the "forgetter" (see section 3) will be placed.

The optional variables `%PRE_CONFIG%`, `%POST_CONFIG%` and `%PRE_FORGET%` are the names of scripts to be run before and after configuration, and before forgetting respectively. The default values of these variables are empty strings.

Some variables are just conveniences to simplify subsequent definitions.

Some string variables, such as `%tlroot%` and `%version%`, can be used outright because they are already set when the launcher starts parsing the ini file. Environment variables, *e.g.* `%appdata%` or

[1] If during installation non-default options are selected for file associations or path adjustment, then a second, modified copy will be written to `texmf-var/web2c`.

%UserProfile% can also be used outright. See the package documentation for the full list.

A few example string definitions:

```
[Strings]
TLNAME=TeX Live %VERSION%
; tlaunch configuration directory
TLCONFIG=%userprofile%\.texlive%VERSION%\tlaunch
TLSCRIPTS=%tlroot%\scripts
POST_CONFIG=%TLSCRIPTS%\post_config.cmd
; optional announcement text
ANNOUNCE=TeX Live Launcher with extras
```

2.2 Filetype associations

In Windows, the association of a filename extension with a program is indirect: an extension is associated with a filetype and a filetype is associated with a command. An example of a filetype definition in the ini file:

```
[FT:TL.TeXworks.edit.%VERSION%]
COMMAND="%tlroot%\bin\win32\TeXworks.exe"
EXTENSIONS=.tex .cls .sty
```

The name of the ini file section consists of the filetype name with an 'FT:' prefix. When a file with a listed extension is double-clicked in a file manager, Windows will run COMMAND with the (quoted) filename appended. If a more complex command is required, *e.g.* with parameters coming after the filename, the section can define a more complex command-line with a SHELL_CMD entry.

2.3 Menus and buttons

The ini file can contain a buttons section and sections for menus, with the latter indicated by an MN: prefix to the section name. Within the section entries, the key is the string to be displayed and the value is the action to be taken.

In the case of a button, the display string is put underneath the button (see figure 1). The launcher tries to find a suitable icon to place on the button itself, but has a fallback icon if it cannot find anything. This fallback icon is used for the Quit button in figure 1.

A few examples:

```
[MN:File]
Browse installation=explorer.exe "%tlroot%\.."
Quit=FU:quit

[MN:Viewers]
PostScript Viewer=FT:TL.PSView.view.%VERSION%
DVI Viewer=FT:TL.DVIOUT.view.%VERSION%

[MN:Documentation]
LaTeX Introduction=SO:%tlroot%\...\lshort.pdf
FAQ=SO:%tlroot%\...\newfaq.pdf

[Buttons]
LaTeX Editor=FU:default_editor
Select default editor...=FU:editor_select
Quit=FU:quit
```

The value, which is the associated action, can take several forms:

- No prefix: a command to be executed.
- With a prefix FT:, the associated action is the COMMAND of the indicated filetype, which should be defined earlier in the file.
- Prefix SO: (shell object) meaning in this case a file or url that Windows should know how to open.
- Prefix SC: indicates a script object defined earlier in the ini file; see the package documentation.
- Prefix FU: indicates a predefined function; see the package documentation.

2.4 The General section

The most important options in this section replicate options from the TEX Live installer:

Filetypes Allowed values are none, new (default) and overwrite

searchpath Allowed values are 0 and 1 (default)

Both entries and the section itself are optional. For example:

```
[General]
FILETYPES=new
SEARCHPATH=1
```

3 Forgetting

The launcher has functions to undo and redo configuration, which can be assigned to menu items. However, the installation may not be under the user's control and may no longer be around when the configuration is to be cleared out.

Therefore, the launcher creates a so-called forgetter as part of its first-time initialization. This forgetter consists of a copy of the launcher and a modified copy of the configuration file, both placed under the user's profile. This copy knows from its location that it is intended to run as forgetter and not as launcher.

4 Scripts

The launcher can run scripts and command-line utilities, and display their output in a window. The ini file can specify scripts for *e.g.* supplemental initialization and cleanup (see section 2.1). Section 6.1 shows some examples. It is also possible to assign scripts to menu entries and to buttons. More about scripts is in the package documentation.

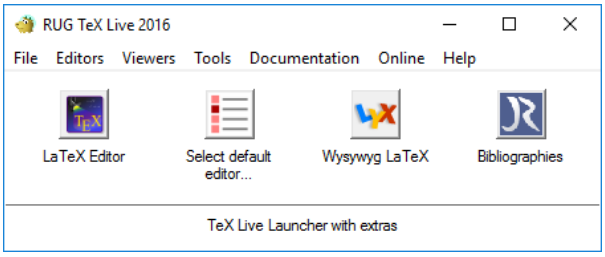

Figure 3: The TEX Live Launcher at the Rijksuniversiteit Groningen

5 Launcher-based installations

The 2017 TEX Live installer offers the option of creating a launcher-based installation, as an alternative to creating menu shortcuts. If this option is selected, then no path adjustment is done and no filetype associations are created by the installer itself. The installer invokes the launcher with a special option to 'install' itself, *i.e.* to create a start menu shortcut and an uninstaller registry entry for itself. In case of a single-user install, it also performs a first-time initialization.

With such an installation, the TEX Live installer no longer has direct dealings with the Windows API, or with the Perl modules providing API access.

For purposes of trying out the launcher, TEX Live includes a script `tlaunchmode` which can switch the installation between classic and launcher mode without reinstalling TEX Live.

6 The launcher at the Rijksuniversiteit Groningen

Workstations at our university are mostly centrally managed. Typically, users have a centrally managed Start menu on their Windows workstation. The IT people put the TEX Live Launcher in this menu, so users are just one click away from starting to use TEX Live.

Settings are centrally backed-up on logout and restored on login. So a user's desktop looks very similar, whatever physical workstation [s]he works on. This same desktop is also available remotely. In addition, users have a network share for storing their own files. This share is also available from any workstation on which they log in.

6.1 Additional software

The additional programs at our university include:

- More editors: TeXnicCenter and TeXstudio. Both offer extensive assistance in editing math.
- The PDF viewer SumatraPDF. This viewer provides source–PDF synchronization for TeXnicCenter, which has no built-in PDF viewer.
- The Java-based bibliography manager JabRef.
- The `epspdf` GUI with bundled single-file Tcl/Tk runtime (`https://ctan.org/pkg/epspdf`).
- The pseudo-WYSIWYG LyX LATEX editor.

There are menu items for additional documentation, such as the LATEX classes for the university house style. Controls for the TEX Live Manager and for uninstalling TEX Live itself are omitted, since those tasks are reserved for the maintainer of the installation.

All these programs were installed on a scratch system and from there copied into the TEX Live installation tree. 'Installed' this way, most of them run more or less ok from their new location.

However, some fixes were desirable and were implemented via a postconfig script (see section 2.1):

TeXnicCenter While TeXnicCenter can autoconfigure itself nicely for MiKTEX, it asks TEX Live users a series of questions about what is where. To spare users those questions, I wrote a vbscript which emulates the MiKTEX autoconfiguration for TEX Live, and which is invoked by the postconfig script.

TeXstudio This editor by default checks at startup whether there is a new version. The postconfig script turns this option off in an existing or newly-created TeXstudio configuration file.

TEXworks borrows some dictionaries from TeXstudio.

SumatraPDF This PDF viewer also tests for updates, which are dealt with in the same way as for TeXstudio. It also requires a registry setting to specify that this is *not* a portable installation, and it should store its settings under the user's profile.

LyX First-time initialization can take a very long time. Therefore, a LyX user configuration directory has been prepared in advance. The postconfig script copies it to the user's profile.[2]

The launcher documentation contains a file `rug.zip` with slightly sanitized versions of the scripts and configuration files actually used at our university installation.

7 Problems

7.1 Non-roaming filetype associations

In a standard Roaming Profiles setup, filetype associations do not roam. I plan to add an option to the launcher to restore missing filetype associations

[2] There is also a shared LyX configuration file which had to be patched, but this is not a task for a per-user postconfig script.

on login. This is not a problem with the centrally-managed desktops at our university. On the other hand, on those centrally-managed desktops some filetype associations are pre-empted and cannot be permanently changed. This includes PDF files.

7.2 Search path

Another problem associated with our desktop management software is that programs ignore the user search path.

This is not a problem for software started from the launcher.

Some programs do not absolutely need TEX Live on the search path. Others, such as TEXworks and the DVI- and PostScript viewers included in TEX Live, are invoked via a wrapper which takes care of the search path. But for TeXstudio I had to provide a wrapper myself to take care of the search path.

And then the university offers software such as R and WinEdt which also need LATEX on the search path, but which are not under my control; the IT department has to handle these.

7.3 Uninstalling

The third problem I want to mention is uninstalling under Windows 10. This is not specific to the launcher.

There are two ways to give a user access to an uninstaller:

- Via a Start menu item. However, Windows 10 may somehow decide not to display such an item.
- Via an uninstaller registry key. This way, it will show up in Settings / Apps / Apps & features. However, Windows may decide to pop up a User Account Control (UAC) prompt even if it is a user install. Still, a user-installed launcher *can* be uninstalled via right-clicking its icon under the Start menu, or from within the launcher itself.

8 Finding out more

Earlier, I mentioned the tlaunch manual. If you have a fully updated 2016 or a later installation of TEX Live with Windows platform support, then you should have the tlaunch binary and documentation on your system. But you can also visit its CTAN directory at `https://ctan.org/pkg/tlaunch`.

For experimentation, you can run the script `tlaunchmode` mentioned at the end of section 5. With this script you can switch an existing TEX Live installation to launcher mode and back.

◇ Siep Kroonenberg
Groningen
The Netherlands
`siepo (at) cybercomm dot nl`