tlcontrib.metatex.org: A complement to TeX Live

Taco Hoekwater

Abstract

TLContrib is a distribution and associated web site that hosts contributed, supplementary packages for TeX Live.

The packages on TLContrib are those not distributed inside TeX Live proper for one or several of the following reasons: because they are not free software according to the FSF guidelines, because they contain an executable update, because they are not available on CTAN, or because they represent an intermediate release for testing.

Anything related to TeX that cannot be in TeX Live but can still legally be distributed over the Internet can have its place on TLContrib.

Keywords: TeX Live, TLContrib, distribution, contribution, packages

1 Introduction

Many readers will be familiar with TeX Live as an easy way to install TeX. This distribution provides a comprehensive TeX system with binaries for most flavors of Unix, including GNU/Linux and Mac OS X, and also Windows. It includes all the major TeX-related programs, macro packages, and fonts that are free software, including support for many languages around the world. The current version is TeX Live 2010.

TeX Live is distributed on DVD by most of the local TeX user groups, but it also allows for continuous package updates over the Internet using the `tlmgr` program.

TeX Live is a wonderful tool, but there are a few considerations to be aware of:

- it contains only FSF-defined 'free software' packages
- it uses CTAN as its primary source for packages
- it does not make interim executable updates
- it is not a suitable medium for package test releases

Each of these limitations has a perfectly reasonable cause:

- The TeX Live maintainers agree (at least for the purposes of working on TeX Live) with the principles and philosophy of the free software movement. Therefore they follow the FSF guidelines on licensing.
- It is good for the TeX community if CTAN is as complete as possible. That gives users one place to look, for instance. Also, it makes it more likely for separate distributions like TeX Live and MiKTeX to be consistent with each other. By using CTAN as the primary package source, TeX Live promotes the use of CTAN.

  A secondary reason for the use of CTAN is that creating a large distribution like TeX Live takes a lot of work, and the number of volunteers is limited. Having a single place to check for new package updates is a lot easier, because this process can be automated to a large extent. Using many separate sources would make this task much more complicated.
- TeX Live ships binaries for 19 different computer platforms, and something like 300 binaries need to be compiled for each of those. Coordinating the task of preparing these binaries is a major effort.
- Because TeX Live is not just a network installation, but also shipped on DVD, it is important that the included packages and binaries are as stable as possible. After all, there is no guarantee that the DVD users will *ever* update their system after the initial installation.

Nevertheless, the limitations of TeX Live mean that there is room for extension. This is the reason for the existence of TLContrib.[1]

On TLContrib, anything that is freely distributable is acceptable, so packages that are not on CTAN are fine, and TLContrib can and will contain updates to executables (though not necessarily for all platforms).

This is possible because the two major limitations of TeX Live do not exist in TLContrib. Firstly, TLContrib is a network-only distribution without the limitations introduced by the physical medium. Secondly, the problem of lack of human resources is solved by offloading the burden of creating and maintaining packages to the actual package maintainers.

Before going on to explain how to use TLContrib, it is important to note the following:

- TLContrib is *not* a full TeX Live repository: it is a complement and contains only its own packages. This means TLContrib can only be used as a secondary repository on top of an existing TeX Live installation.
- TLContrib is not maintained by the TeX Live team: the responsibility for the actual packages lies with the package maintainers themselves,

Editor's note: This article appeared originally (in slightly different form) in *MAPS* 41, fall 2010.

[1] The web site for TLContrib is `http://tlcontrib.metatex.org/`

and the server maintenance is handled by yours truly.

There is no competition between TLContrib and TEX Live, but as one of the goals of TLContrib is to ease the workload of the TEX Live team, it would not make much sense for them to be the actual maintainers. For this reason there is a separate mailing list dedicated to TLContrib.[2] Please address your questions related to packages obtained from TLContrib there, and not on the regular TEX Live list.

2 Using TLContrib as a distribution

First things first: before attempting to use TLContrib, make sure that you have the latest (network) update of TEX Live 2010, and in particular that you run the latest `tlmgr`. During the development of TLContrib, a small number of incompatibilities were found in the `tlmgr` as distributed on the DVD that have since been fixed. Furthermore, the current version of TLContrib works only with TEX Live 2010 and not for any earlier versions of TEX Live.

And a warning: Executable packages are not necessarily available for all platforms on TLContrib. Unfortunately, it appears that the current TEX Live update manager is not smart enough to correctly detect versioning in dependencies. In practice, this means that you should not update packages that depend on executable package updates *unless* the actual executable package update is also available on TLContrib for your platform.

Keeping the above in mind, in order to use TLContrib as an extra repository in the TEX Live 2010 package manager (`tlmgr`), there are two options, depending on whether you prefer to use the command line version or the GUI version of the TEX Live 2010 package manager.

2.1 Graphical interface usage

In the GUI version of the package manager, select the menu item Load other repository ... from within the tlmgr menu. Set the value to

```
http://tlcontrib.metatex.org/2010
```

There is currently no way to save this setting.

Besides not being able to save the TLContrib setting, when using the graphical user interface it is not always easy to see whether executable package updates are available. For this reason you should consider using the command line version of the package manager for use with TLContrib, even if you are accustomed to using the GUI interface.

[2] The mailman page for the mailing list is `http://www.ntg.nl/cgi-bin/mailman/listinfo/tlcontrib`.

2.2 Command line usage

The simplest approach is to just start `tlmgr` from the command line with an extra option:

```
$ tlmgr --repository \
  http://tlcontrib.metatex.org/2010
```

If you plan to use TLContrib regularly, it makes sense to define a shell alias to save you some typing (this trick is courtesy of Will Robertson). To do so, put the following into your `.bash_profile` or equivalent (this has to be on a single line):

```
alias tlc="tlmgr --repository
          http://tlcontrib.metatex.org/2010"
```

You can then view what is available in the TLContrib repository with standard `tlmgr` commands such as

```
$ tlc list
```

to see what is currently available for installation. Packages can be updated to their pre-release versions by typing, say,

```
$ tlc update siunitx
```

If an update performed in this way 'goes bad' and you'd like to revert to the official release, execute

```
$ tlmgr install siunitx --reinstall
```

and things will be back to normal.

3 Using TLContrib for distribution

The rest of this article describes further details important for a package maintainer aiming to use TLContrib for distribution.

Before you decide to add a package to TLContrib, please bear this in mind:

- It is not the intention of TLContrib to replace either TEX Live or CTAN: if a package is not blocked from TEX Live for one of the reasons mentioned earlier, and can be made available on TEX Live or CTAN, then it should not be part of TLContrib at all.

In order to be able to upload packages to TLContrib, you have to be a registered user. You can register as a user via the TLContrib web site, and, not by coincidence, this is also the place where you create new packages and package releases.

After registration is complete, you can log into TLContrib by following the member section link.

If you do upload a package to TLContrib, please also subscribe to the TLContrib mailing list, because any questions about your package are likely to be made there.

3.1 Package creation example

This quick start guide uses an update of the
  `context-lettrine`

package as an example of how to create a package. In the following, you need to replace `context-lettrine` by the actual package name that *you* are updating, of course.

Besides being logged in to TLContrib, the first thing you need to do is to create your updated package source. In this case, the easiest way is to start from the current TeX Live version, so first you have to fetch the current `context-lettrine` archive(s) from the network distribution of TeX Live. The base URL is: `http://www.ctan.org/tex-archive/systems/texlive/tlnet/archive`.

In fact, for this example, there are two archives to be downloaded:

```
context-lettrine.tar.xz
context-lettrine.doc.tar.xz
```

For some TeX Live packages there is even a third archive file named ⟨*package*⟩`.source.tar.xz`. This is because the distribution system of both TeX Live and TLContrib splits the contribution into run-time files, documentation files, and source files. Users can ask the installer not to install the last two file types to save on disk space and network traffic.

You have to create a single local archive file with the combined and updated content of the two downloaded archives. After extracting both `tar.xz` files in the same directory, you will have a tree structure that looks like this:

```
doc/
  context/
    third/
      lettrine/
        lettrine-doc.pdf
        lettrine-doc.tex
        W.pdf
tex/
  context/
    interface/
      third/
        lettrine.xml
    third/
      lettrine/
        t-lettrine.tex
tlpkg/
  tlpobj/
    context-lettrine.doc.tlpobj
    context-lettrine.tlpobj
```

First, delete the whole `tlpkg` sub-tree. The `tlpobj` files contain meta-data specific to each particular revision of a package, and the information in the downloaded version of these files will henceforth be no longer applicable. New versions of the `tlpobj` files will be generated automatically by TLContrib's distribution creation tool.

You may now update the other files in the tree, and create the archive file (the acceptable formats are `tar.gz`, `tar.xz`, and `zip`). Please read the next section, 'About package sources', carefully before finalizing the archive.

The TLContrib `context-lettrine` package will use the newly created archive as source for the package, so make doubly sure you use the right files. Starting with the existing TeX Live package archive(s) is just so you get an idea of what goes where: sometimes TeX Live packages contain more files and symbolic links than you initially expect. You can build the source package completely from scratch if you want, but it is easy to forget files if you don't check.

Incidentally, while the base name of the local archive file does not matter, you have to make sure that the extension is `.tar.gz`, `.tar.xz`, or `zip`, otherwise the upload will fail.

Now go to `http://tlcontrib.metatex.org` in your browser, log in, and click new package. As the new package is an update to TeX Live, make sure you select that option, and the proper package name from the drop-down (`context-lettrine`).

In the next screen, most of the needed input will be automatically filled in for you, based on the current TeX Live revision of the package.

Edit the rest of the input form to have a proper version and set the source to File upload. Its value has to be set to the new archive that was created earlier. Adjust the Release state drop-down so it is set to public. It is also wise to check the license field, for it does not always import correctly due to database mismatches.

Then press submit new revision, verify the upload, and submit again to finalize the new package.

Assuming all went well, all that is needed for now is to wait until the hour has passed: your package should be available from the TLContrib repository after that.

The TLContrib distribution system works asynchronously: the front-end data that you as a package maintainer can create and modify is exported to the user-side TLContrib repository by a cron job that runs independent of the actual web site. Currently this cron job runs hourly, on the hour.

3.2 About package sources

Please note: Currently only the `tar.gz`, `tar.xz`, and `zip` archive formats are supported in the File upload and HTTP URL methods, and there are further strict requirements on the archive itself:

For a non-executable package, it should contain a complete TDS (TeX Directory Structure; see `http://www.tug.org/tds/`) sub-tree. In TeX Live, normally all macro files go under `texmf-dist`, and, in that case, this directory level can be skipped in the archive (it will be added automatically by the TLContrib publication system). Be advised that, in general, uploading a CTAN zipped folder will *not* work, because CTAN packages are almost never in TDS format.

For an executable package, you can also use the TDS layout (with the binaries in `bin/$ARCH/`), but if you only have files inside the binaries folder, you can skip the directory structure completely: in this case, the TLContrib publication system will automatically add the needed structure.

Make sure that your archive contains only files that belong to your package, and especially that it does not accidentally overwrite files owned by other packages.

Also, check twice that the archive contains only files that belong in the TDS: Delete backup files, and remove any special files that may have been added by the operating system (Mac OS X especially has a very bad habit of adding sub-directories for its Finder that really do not belong in the package).

It is not always simple to guess what should go into a TeX Live update package. If you are building such an updated package, it is always wise to start from the existing TeX Live sources.

TLContrib accepts no responsibility for package contents: the system does run some sanity checks, but ultimately, you as maintainer are responsible for creating a correctly functioning package. Badly behaving or non-working packages will be removed on executive decision by the TLContrib maintainer(s) without prior notice.

3.3 Package creation in detail

When you create a new package, a short wizard will help present itself to help you set up the package type. There are two types of packages: those that are updates of existing TeX Live packages, and those that are standalone. The wizard screen presents you the choice between these two types, and a drop-down listing TeX Live packages. The list of existing TeX Live packages is updated daily. Once this decision is made, it becomes fixed forever: the Id field of a package cannot be edited afterwards.

The Id field is the internal identifier of the package. An Id should consist of a single 'word', with a length of at least two characters, that contains only alphanumerics, dashes, and underscores. It can optionally followed by a platform identifier, which is then separated from the first part by a single dot.

Also note that when Release state becomes public (as explained below), it will no longer be possible to edit that particular release of the package. All further edits will force the creation of a new release, with a new revision id, and needing new sources.

Yet another note: If you intend to create an executable package, you have to be really sure you know what you are doing. Creating portable binaries for any platform is far from trivial. Paraphrasing Norbert Preining from the TLContrib mailing list:

> If you have NO experience with compiling, preparing binaries for various platforms, distributing, etc., JUST DO NOT GO THERE!

Macro packages are much easier; for those you need only a good understanding of how the TDS works.

3.4 Package editing

After the initial New package wizard screen, or after pressing Edit in the your package list for pre-existing packages, you will be presented with a fairly large edit screen.

During the initial TLContrib package creation process, if the package is updating an existing TeX Live package, certain fields will have been filled in automatically from the TeX Live package database. Otherwise you will have to fill in everything yourself.

Title
: The human-readable name of your package.

Description
: This is a description in a few sentences of what the package does.

Package type
: Even though the drop-down is long, really there are only two choices in the drop-down: A package is either a Macro package, or na Executable package. The distinction is important because the required package source structure is different for each of the two types, as explained below.

TLMGR directives
: A list of `tlmgr` directives, e.g. addMap or addFormat. A better interface is planned, but, for the moment, you have to make sure you know what you are doing. Have a look at the existing TeX Live package database (the file named `texlive.tlpdb`) for examples.

  You only have to specify the directives; do not add execute at the start.

TL dependencies
: Package Ids of other TeX Live packages on which this package depends, one per line. Unless you know exactly what is needed, it is probably best to leave this field blank, but in any case:

  You only have to specify the package Ids; do not add depend at the start. If your package depends on an executable package, for example luatex, write the Id as luatex.ARCH. Doing so will make `tlmgr` automatically select the appropriate executable package for the user's platform.

TL postactions

A list of `tlmgr` post-install actions, e.g. shortcut or fileassoc. A better interface for this is also planned, but, for the moment, you have to make sure you know what you are doing here as well. Have a look at the existing TEX Live package database (`texlive.tlpdb`) for examples.

You only have to specify the actions, do not add postaction at the start.

License

Pick one from the two drop-downs, and set the radio button accordingly. If you need to use Other free license or Other non-free license, please drop me an email. I am sure the list is incomplete. In this context, Free means: according to the Debian Free Software Guidelines.

Log message

This field is just for release notes: it will not be exported to the TLContrib repository. The SVN URL and GIT URL methods will automatically refill in this field with the remote revision and log message. For other source methods, you can fill in whatever seems appropriate.

Release state

Only packages that are public are exported, but this also has side-effects. Once the Release state is public, it is no longer possible to edit a package release on the spot. Submitting the form in that case will always create a new release.

On edits, you will see some extra information: Synch state and rev. The first is the current status of a package release with respect to the published repository, the second is the revision number that has been assigned to this release.

Version

This is the user-visible version field.

Source

Here things get interesting. There are five ways to put the source of a package release into the database, as explained in the next sections.

- As previous revision
  If you are editing an already existing package, then it is possible to re-use the uploaded source from the revision you are editing as the source for the new revision that will be created.
- File upload
  Upload of a local archive file via CGI. Be warned that if there are other errors in your form, you will have to re-select the local file after fixing those other errors. Contrary to the other fields, local file selection is not persistent across form submits.
- HTTP URL
  This asks the system to do a `wget` of an archive file on a specific URL, which could be either HTTP or FTP. If you need remote log-in information to access the file, please encode the user name and password in the URL, exactly as you would do when using `wget` on the command line.
- SVN URL
  This asks the system to do an `svn checkout` of a specific URL. In this case, you may also need SVN Username and SVN Password. Also, some repositories may need `anonymous` as the user name for anonymous access. The top-level checkout folder will be stripped away before creating the package. This is so that you can give, e.g. `http://foundry.supelec.fr/svn/metapost/trunk/texmf/`, as the URL without getting an extra directory level.
- GIT URL
  This asks the system to do a `git clone` of a specific URL. It is very similar to SVN URL, just using a different versioning system. In this case, you may also need GIT Branch.

Please verify package contents

The first time the edit form is loaded, this will only display a message, but after the initial submit (assuming everything else went well), it will display the full list of files that will become the source of your package.

Please check this list carefully! TLContrib does run some tests on the package contents and will refuse to accept package sources that are horribly wrong, but it does not check the actual contents of any of the files, and of course it cannot test for every possible problem.

3.5 Package transfer

It is possible for the maintainer of a package to transfer the package to another user completely. To do so, follow the Share link in your package list. See the help text in that form for details.

3.6 Package sharing

It is also possible for the maintainer of a package to share the package maintenance with other users.

To set up package sharing for a package you maintain, follow the Share link in your package list. See the help text in that form for details.

When someone else has shared a package with you, then you will see new entries in your package list. These will have the user id of the actual package maintainer added after the Date field. You can edit such packages (and thus create new revisions), but the new revisions will become property of the actual package maintainer.

In other words: a package can have only one actual maintainer, and that maintainer is responsible for all revisions of the package. However, the maintainer can allow other users to help with the actual creation of new revisions.

3.7 Package deletion

In the list of your packages and in the view screen of one of your package releases, there are two links that delete items:

- Del / Delete revision

  This link deletes a single revision of a package.
- Delete package (in view screen only)

  This link removes a whole package completely, including all revisions of it.

Both links show a confirmation screen first.

3.8 Remote revision creation (advanced usage)

Once a package has been created (there must be at least one revision record present already), and under the conditions that it has a source method of HTTP URL, SVN URL, or GIT URL, it is possible to submit a new revision by fetching a special URL from a remote location or script. Using this method, there is no need to be logged in at all.

The URL template looks like this:

```
http://tlcontrib.metatex.org
   /cgi-bin/package.cgi/action=notify
   /key=⟨key⟩
   /check=⟨md5⟩
   ?version=⟨version⟩
```

Please note that `version` is preceded by a question mark, but everything else is separated by slashes, and, of course, the actual URL should be a single line, without any spaces. All three fields are required. The three special fields have to be filled in like this:

⟨*key*⟩

This is the Id of the package. For example, we'll use `luatex.i386-linux`.

⟨*md5*⟩

This is a constructed checksum, created as follows: it is the hexadecimal representation of the md5 checksum of the string created by combining your userid, your password, and the new version string, separated by slashes.

For example, let's assume that your userid is `taco` and your password is `test`, and that the new release that you are trying to create has version `0.64.0`. On a Unix command line, the checksum can then be calculated like this:

```
$ echo taco/test/0.64.0  md5sum
c704f499e086e0d54fca36fb0abc973e  -
```

The value of ⟨*md5*⟩ is therefore:

```
c704f499e086e0d54fca36fb0abc973e
```

⟨*version*⟩

This is the version field of the new release.

Note: if this contains spaces or other characters that cannot be used in URLs, then you either have to escape the version string in the URL, or use POST instead of GET. In any case, do not escape the version while calculating the checksum string.

There is no need to do any URL escaping in our case, so our value of ⟨*version*⟩ will be `0.64.0`.

Using the example variables given above, the final URL that would have to be accessed is (again without line breaks or spaces):

```
http://tlcontrib.metatex.org
  /cgi-bin/package.cgi/action=notify/
  key=luatex.i386-linux
  /check=c704f499e086e0d54fca36fb0abc973e
  ?version=0.64.0
```

Accessing this URL will cause TLContrib to fetch the HTTP or SVN or GIT URL source in the package's top-level revision (regardless of what its publication state is), and create a new revision based on the fetched file(s) and the supplied version string. All other fields will remain exactly the same as in the original top-level revision.

This new package revision will appear in the web interface just like any other revision, there is nothing special about it other than what is already mentioned.

4 Final remark

TLContrib is a fairly new project, and some improvements are definitely possible, especially in the edit forms on the web site. But I hope that even in the current state, it will be a useful addition to the whole TeX Live experience.

◇ Taco Hoekwater
http://tlcontrib.metatex.org