

T_EX Live's new infrastructure

Norbert Preining

Vienna University of Technology, Austria

GU_Tmeeting 2007

Pisa, Italia 13 October 2007

Properties of the T_EX Live distribution

- ▶ includes all the free stuff from CTAN
- ▶ ready for “consumption”, i.e., runs from DVD, but can also be installed into the file system
- ▶ available for a wide range of platform–operating system combinations
- ▶ currently is replacing teT_EX in many (Unix) distributions as default T_EX system

Upstream organization

- ▶ SVN repository where many people have write permissions
- ▶ loads of supporting scripts doing a variety of jobs:
 - ▶ preparing the installation for mastering
 - ▶ installation from various media
 - ▶ installation of packages from CTAN into the SVN repository
 - ▶ performing various checks on the whole archive (coverage, double inclusion, etc.)

Upstream organization

- ▶ SVN repository where many people have write permissions
- ▶ loads of supporting scripts doing a variety of jobs:
 - ▶ preparing the installation for mastering
 - ▶ installation from various media
 - ▶ installation of packages from CTAN into the SVN repository
 - ▶ performing various checks on the whole archive (coverage, double inclusion, etc.)

Problems

- ▶ hand-crafted, hard to read
- ▶ not easily extendible
- ▶ no documentation

Previous infrastructure: tpms

T_EX Package Manage[r/ment] files collected a variety of information into on XML file:

- ▶ file patterns and file lists
- ▶ title, description, license, versions
- ▶ activation of map files, formats, hyphenation patterns

Previous infrastructure: tpms

T_EX Package Manage[r/ment] files collected a variety of information into on XML file:

- ▶ file patterns and file lists
- ▶ title, description, license, versions
- ▶ activation of map files, formats, hyphenation patterns

Problems with tpms

- ▶ mixture of generated and static information
- ▶ duplicated information (version, license, descriptions, ...) which were outdated
- ▶ hard to parse, thus the necessity to generate another set of files for the installer

Aims of the new infrastructure

- ▶ Separation of static from generated content
Going from a ‘source’ to the ‘object’ should include automatically data from various other sources (mainly the Catalogue and the repository)

Aims of the new infrastructure

- ▶ Separation of static from generated content
Going from a ‘source’ to the ‘object’ should include automatically data from various other sources (mainly the Catalogue and the repository)
- ▶ Getting rid of any ‘additional’ files (list files)
any additional files tend to be outdated, all the necessary information should be present in *one* place and be easily parseable.

Aims of the new infrastructure

- ▶ Separation of static from generated content
Going from a 'source' to the 'object' should include automatically data from various other sources (mainly the Catalogue and the repository)
- ▶ Getting rid of any 'additional' files (list files)
any additional files tend to be outdated, all the necessary information should be present in *one* place and be easily parseable.
- ▶ Single package updates via the web
updates to single packages (e.g., a new beamer release)

Aims of the new infrastructure

- ▶ Separation of static from generated content
Going from a 'source' to the 'object' should include automatically data from various other sources (mainly the Catalogue and the repository)
- ▶ Getting rid of any 'additional' files (list files)
any additional files tend to be outdated, all the necessary information should be present in *one* place and be easily parseable.
- ▶ Single package updates via the web
updates to single packages (e.g., a new beamer release)
- ▶ Better documentation
since T_EX Live is replacing teT_EX we want to give people incorporating T_EX Live into distributions a better documented and easier to handle system

The new infrastructure – the basics

Three types of objects/files:

- ▶ `tlp_src`: provides only the minimal information
- ▶ `tlp_obj`: the extension/compilation of a `tlp_src` file which includes the complete file lists and all the additional information derived from other sources
- ▶ `tlp_db`: the collection of all available/installed/upgradeable `tlp_obj`s

tlp_src format by example

beamer.tlp_src

name_beamer

category_Package

depend_Package/pgf

tlp_src format by example

beamer.tlp_src

```
name_beamer
category_Package
depend_Package/pgf
```

bin-cweb.tlp_src

```
name_bin-cweb
category_TLCore
docpattern_f_texmf/doc/man/man1/ctangle.1
docpattern_f_texmf/doc/man/man1/cweave.1
docpattern_f_texmf/doc/man/man1/cweb.1
binpattern_f_bin/${ARCH}/ctangle
binpattern_f_bin/${ARCH}/cweave
```

Allowed fields for `tlpsrc`

- ▶ `name`: identifies the package
- ▶ `category`: one of (currently) `Collection`, `Scheme`, `TLCore`, `Documentation`, `Package`
- ▶ `catalogue`: link to the T_EX Catalogue
- ▶ `shortdesc`, `longdesc`: description of the package
- ▶ `depend`: `Category/Name` (multiple entries possible)
- ▶ `execute`: activating maps, formats, hyphenation patterns
- ▶ `srcpatterns`, ...: selecting member files of the package via patterns

The pattern language

patterns are of the form

[PREFIX]TYPE PAT

where PREFIX can be +, !+, or !,

The pattern language

patterns are of the form

[PREFIX]TYPE PAT

where PREFIX can be +, !+, or !, and TYPE PAT can be:

f path includes all files which match `path` where *only* the last component of `path` can contain the usual glob characters * and ? (but no others!).

The pattern language

patterns are of the form

[PREFIX]TYPE PAT

where PREFIX can be +, !+, or !, and TYPE PAT can be:

- f path** includes all files which match `path` where *only* the last component of `path` can contain the usual glob characters * and ? (but no others!).
- d path** includes all the files in and below the directory specified as `path`.

The pattern language

patterns are of the form

[PREFIX]TYPE PAT

where PREFIX can be +, !+, or !, and TYPE PAT can be:

f path includes all files which match `path` where *only* the last component of `path` can contain the usual glob characters * and ? (but no others!).

d path includes all the files in and below the directory specified as `path`.

t word₁ ... word_N word_L includes all the files in and below all directories of the form

word₁/word₂/.../word_N/.../any/
dirs/.../word_L/

The pattern language

patterns are of the form

[PREFIX]TYPE PAT

where PREFIX can be +, !+, or !, and TYPE PAT can be:

f path includes all files which match `path` where *only* the last component of `path` can contain the usual glob characters `*` and `?` (but no others!).

d path includes all the files in and below the directory specified as `path`.

t word1 ... wordN wordL includes all the files in and below all directories of the form

`word1/word2/.../wordN/.../any/
dirs/.../wordL/`

r regexp includes all files matching the Perl regexp `/^regexp$/`

Example patterns

- ▶ `runpattern f texmf/chktex/*`
includes all files *in* `Master/texmf/chktex/`

Example patterns

- ▶ `runpattern f texmf/chktex/*`
includes all files *in* `Master/texmf/chktex/`
- ▶ `binpattern f bin/${ARCH}/bibtex`
includes the bibtex binaries into the bin-bibtex package,
depending on the architecture

Example patterns

- ▶ `runpattern f texmf/chktex/*`
includes all files *in* Master/texmf/chktex/
- ▶ `binpattern f bin/${ARCH}/bibtex`
includes the bibtex binaries into the bin-bibtex package, depending on the architecture
- ▶ `runpattern d texmf/tex/lambda/base`
includes all files in and under the above path

Example patterns

- ▶ `runpattern f texmf/chktex/*`
includes all files *in* Master/texmf/chktex/
- ▶ `binpattern f bin/${ARCH}/bibtex`
includes the bibtex binaries into the bin-bibtex package, depending on the architecture
- ▶ `runpattern d texmf/tex/lambda/base`
includes all files in and under the above path
- ▶ `runpattern t texmf-dist/omega/uni2char`
includes all files in texmf-dist/omega/.../uni2char/

Example patterns

- ▶ `runpattern f texmf/chktex/*`
includes all files *in* `Master/texmf/chktex/`
- ▶ `binpattern f bin/${ARCH}/bibtex`
includes the bibtex binaries into the bin-bibtex package, depending on the architecture
- ▶ `runpattern d texmf/tex/lambda/base`
includes all files in and under the above path
- ▶ `runpattern t texmf-dist/omega/.../uni2char`
includes all files in `texmf-dist/omega/.../uni2char/`
- ▶ `runpattern r */foobar`
includes the files matching the regexp

Autogenerated patterns

To keep `tlp_src` files small, if a pattern section is empty or all patterns are prefixed with `+`, the following patterns are automatically generated:

- ▶ `runpatterns` in category `Package`:

```
t texmf-dist topdir name
```

- ▶ `docpatterns` in category `Package`:

```
t texmf-dist doc name
```

- ▶ `docpatterns` in category `Documentation`:

```
t texmf-doc doc name
```

- ▶ `srcpatterns` in category `Package`:

```
t texmf-dist source name
```

- ▶ `srcpatterns` in category `Documentation`:

```
t texmf-doc source name
```

Additional tricks

arch expansion In case the string `#{ARCH}` occurs in one `binpattern` it is automatically expanded to the respective architecture.

bat/exe/dll for win32 For `binpatterns` of the form `f bin/win32/foobar` files `foobar.bat`, `foobar.dll`, and `foobar.exe` are also matched.

Additional tricks

arch expansion In case the string $\${ARCH}$ occurs in one `binpattern` it is automatically expanded to the respective architecture.

bat/exe/dll for win32 For `binpatterns` of the form `f bin/win32/foobar` files `foobar.bat`, `foobar.dll`, and `foobar.exe` are also matched.

Effects of auto generation and tricks

total number of `tlpsrc` files: 1561

total number of `tlpsrc` files with patterns: 168

number of bin- and hyphen- `tlpsrc` files with patterns: 125

number of 'normal' packages with patterns: 43

tlpobj format by example

Excerpt from bin-dvipsk.tlpobj

```
name_bin-dvipsk
category_TLCore
revision_4427
docfiles_size=959434
  _texmf/doc/dvips/dvips.html
  ...
runfiles_size=1702468
  _texmf/dvips/base/color.pro
  ...
  _texmf/scripts/pkfix/pkfix.pl
binfiles_arch=i386-solaris_size=329700
  _bin/i386-solaris/afm2tfm
  _bin/i386-solaris/dvips
  _bin/i386-solaris/pkfix
binfiles_arch=win32_size=161280
  _bin/win32/afm2tfm.exe
  ...
```

Allowed fields in `tlpobj` files

- ▶ XXXfiles take the place of XXXpatterns
every files section has a size attribute, and the binfiles section can occur more then once with different arch tags (see above)
- ▶ revision: maximum svn revision number of the contained files, since version numbers are not parseable, trustworthy, or not even present
- ▶ catalogue-* keys: stuff taken from the catalogue for example catalogue-version, catalogue-authors, catalogue-license

`tlpdb` format

concatenation of `tlpobj`

Perl programming API

TeXLive::TLTREE properties of the subversion repository, in principle it is `svn status -v`

TeXLive::TeXCatalogue simple interface to the \TeX Catalogue

TeXLive::TLPSRC for `tlpsrc` files, basic functionality like reading in, writing out, and member access functions. Generation of a TLPOBJ object from a TLPSRC object using an instance of TLTREE.

TeXLive::TLPOBJ for `tlpobj` files, basic functionality like reading in, writing out, and member access functions. Generation of zip files for web updates and installation media.

TeXLive::TLPDB access to the \TeX Live database.

TeXLive::TLUtils some handy functions used in all the other modules.

Documentation

- ▶ all modules contain a full documentation in pod format
- ▶ additional text API document
- ▶ the article accompanying this talk

Documentation

- ▶ all modules contain a full documentation in pod format
- ▶ additional text API document
- ▶ the article accompanying this talk

Additional APIs

- ▶ shell api at a very basic level
- ▶ python api

Integration into T_EX Live

- ▶ ctanztl: installs packages from CTAN into T_EX Live repository - updated
- ▶ installer: work has started
- ▶ various cron jobs (updating the database): ready
- ▶ consistency checks: missing
- ▶ all tpm stuff has been removed from the repository, everything new will be written using these modules

Resources

- ▶ `tex-live@tug.org` - main contact point
- ▶ `www.tug.org/texlive` - the main entry point, with links to developers' resources, documentation
- ▶ `www.tug.org/svn/texlive/trunk/` - web view onto the subversion repository;
- ▶ `svn://tug.org/texlive/trunk` - svn repository, anonymous access
- ▶ `www.tug.org/texlive/pkgupdate.html` - an explanation how updates from CTAN to T_EX Live are done.

Resources

- ▶ `tex-live@tug.org` - main contact point
- ▶ `www.tug.org/texlive` - the main entry point, with links to developers' resources, documentation
- ▶ `www.tug.org/svn/texlive/trunk/` - web view onto the subversion repository;
- ▶ `svn://tug.org/texlive/trunk` - svn repository, anonymous access
- ▶ `www.tug.org/texlive/pkgupdate.html` - an explanation how updates from CTAN to T_EX Live are done.

Thanks a lot, and hope to see you around