

Tidydoc Manual

The Documentation Organizer
Edition 0.2, for Tidydoc Version 0.2
6 November 2005

by Nicolas Burrus

This file documents the `tidydoc` command for generating a documentation browser.

Copyright (C) 1992, 1993, 1996, 2002, 2005 Free Software Foundation, Inc.

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions, except that this permission notice may be stated in a translation approved by the Foundation.

Short Contents

1	Overview	1
2	Adding documents	3
3	Reorganizing documents	7
4	Generating HTML	9
5	Generating Bibtex	13

Table of Contents

1	Overview	1
1.1	License	1
1.2	Introduction	1
1.3	Features	1
1.4	Quick start	2
2	Adding documents	3
2.1	Using <code>td-add-doc</code>	3
2.2	<code>.dsc</code> file format	3
2.3	Configuration file	4
3	Reorganizing documents	7
3.1	Using <code>td-reorganize-doc</code>	7
3.2	Additional notes	7
4	Generating HTML	9
4.1	How it works	9
4.2	XML descriptions	9
4.3	Using <code>td-generate</code>	9
4.4	Configuration file	10
4.5	Template files	10
4.6	Using Treeview	11
5	Generating Bibtex	13

1 Overview

1.1 License

This program is licensed under the Gnu GPL license <http://www.gnu.org/copyleft/gpl.html>.

1.2 Introduction

The goal of the `tidydoc` program is to make documentation organization easier. It addresses the following problems:

- Add documents to a pool of documents
- Give them the right names and put them into the right directories
- Generate a set of HTML pages to browse your documents

`tidydoc` is written in Python in a spirit of flexibility and easy tweaking, but with default behaviors which fit the needs of most people.

You can add documents using arbitrary commands, such as a simple `cp` or through `ssh` with `scp`.

`tidydoc` can then generate HTML files indexing your documents. A simple XML file is required for each document, including some informations. These description files may be generated by using `tidydoc` upload facilities, but this is not mandatory.

No additional tool is required, a standard Unix system with a Python interpreter will do the job.

1.3 Features

Version 0.2 includes the following features:

- Documents uploading
 - Simple `.dsc` files can be used instead of XML documents
 - Any upload command can be used
 - Multiple categories are supported
 - Files are automatically renamed using title, author and date information
 - Files are automatically put into the right directories
 - Some parameters are automatically extracted from PDF files
- HTML generation
 - Handy navigation panel using Treeview <http://www.treeview.net>
 - Some directories may be considered as row documentation
 - Flexible output using templates
- Bibtex generation

`tidydoc` was written by Nicolas Burrus nicolas.burrus@ensta.fr.

Many concepts were inspired from Orgadoc <http://www.gnu.org/software/orgadoc/>.

1.4 Quick start

This section describes the simplest way to start using tidydoc. This will be done through a commented sequence of shell commands:

```
$ tar xvfj tidydoc-0.2.tar.bz2 # uncompress the sources
$ cd tidydoc-0.2
$ ./configure --prefix=/usr/local # files will get installed in /usr/local
$ make # generate files and documentation
$ make install # installation, you might need to be root
$ mkdir /var/www/tidydoc
$ td-create-htmlroot /var/www/tidydoc # initialize the html root dir.
$ cp /usr/local/share/doc/tidydoc/examples/tidydoc.conf /etc/ # or /usr/local/etc/
$ emacs /etc/tidydoc.conf # you can replace emacs by your favorite text editor ...
=> set input_path and output_path to /var/www/tidydoc
```

Now you can start adding documents:

```
$ td-add-doc my_favorite_doc.pdf
$ td-add-doc http://www.good_website.org/document.pdf
```

And generating html and bibtex files:

```
$ td-generate
=> /var/www/tidydoc/index.html gives access to your documentation tree
=> /var/www/tidydoc/documents.bib now contains the bibtex entries
```


2 Adding documents

2.1 Using td-add-doc

To add documents, use the `td-add-doc` command.

The format for running the `td-add-doc` program is:

```
Usage: td-add-doc [options] document_file [description_file]
```

Options

```
-c, --config configfile  Config file to use.
-v, --version            Print version number.
```

`'document_file'`

Document to add. The name of the document does not matter.

`'description_file'`

Template to use to create the `.dsc` file. If not specified, `tidydoc` will generate one automatically.

`td-add-doc` open the document and ask you to fill out a `.dsc` file describing its content. A confirmation is required before actually sending the file.

Destination directories and file names will be determined from the description file. If a new category is introduced, a new directory will be automatically created. The naming scheme for files is `'first_author.year.conf.title.extension'`.

2.2 .dsc file format

`.dsc` files are simples. Here is a sample file:

```
% This is a comment
Title: Fractals are everywhere in real world

% One author per line
Author: Patrick Foobar

Date: 1989

Conf: micai

Pages: 4

Keywords: fractals nature

% Url towards the document.
Link: http://patrick.foobar.com/fractals.pdf

% Line breaks are taken into account.
Abstract: Fractional Brownian motions number of physical phenomena
```


fractional order. However, the precise meaning of such complementary approaches are The first one, based on nonstationary nature of measurements; the second one, self-similarity properties of FBM and reveals an underlying stationary structure relative to each time-scaling.

```
% public or private
Visibility: public
```

```
Language: english
```

```
% Kind of bibtex entry
Bibtex type: article
```

```
% Additional bibtex fields
Bibtex fields:
volume = {5}
number = {2}
pages = {121--125}
```

```
Comment:
```

```
% One category per line
Categories:
    math/fractals/papers
    by_author/flandrin
```

For PDF files, some of these fields might be automatically filled by `td-add-doc`.

2.3 Configuration file

Start by using the sample configuration file:

```
# Destination path for doc uploads. May be a directory on a remote host.
upload_dest_path = "/tmp/tidydoc"

# File containing the categories list.
# td-generate creates a categories file into output_path.
# This is used by td-add-doc to list existing categories.
categories_path = output_path + '/td_categories'

# Command which will be executed to upload a document into its first category.
# Some variables will be substituted:
# %(destdir)s : upload_dest_path / category
# %(orig_doc_path)s : path towards the input document
# %(orig_dsc_path)s : path towards the input description
# %(final_doc_name)s : destination name of the document
# %(final_dsc_name)s : destination name of the description
upload_commands = \
```



```

"""
mkdir -p "%(destdir)s" || exit 1
cp "%(orig_doc_path)s" "%(destdir)s"/"%(final_doc_name)s" || exit 1
cp "%(orig_dsc_path)s" "%(destdir)s"/"%(final_dsc_name)s" || exit 1
"""

# Command which will be executed to upload a document into other categories.
# Some variables will be substituted:
# %(destdir)s : upload_dest_path / category
# %(rel_first_destdir)s : path towards the directory of the first category
# %(final_doc_name)s : destination name of the document
# %(final_dsc_name)s : destination name of the description
link_commands = \
"""
mkdir -p "%(destdir)s" || exit 1
ln -sf "%(rel_first_destdir)s/%(final_doc_name)s" "%(destdir)s"/"%(final_doc_name)s" |
ln -sf "%(rel_first_destdir)s/%(final_dsc_name)s" "%(destdir)s"/"%(final_dsc_name)s" |
"""

# Commands which will be executed when a document is to be added.
# The document will then be uploaded.
# The following variables are substituted:
# %(orig_doc_path)s : path of the input document
# %(tmp_dsc_path)s : path of the temporary description file already created
add_doc_commands = \
"""
kfmclient exec "%(orig_doc_path)s" &
emacs "%(tmp_dsc_path)s"
"""

```

Configuration variables are Python instructions. Some special variables will be substituted by tidydoc.

3 Reorganizing documents

3.1 Using `td-reorganize-doc`

Usage: `td-reorganize-doc` [options] `dsc_file`

Options

`-c, --config configfile` Config file to use.
`-v, --version` Print version number.

`td-reorganize-doc` takes a description file, and modify the corresponding document organization accordingly. A typical use case is to modify a `.dsc` file, e.g. changing the title and adding a new category, and then calling `td-reorganize-doc` to rename the file and link it into the new categories.

3.2 Additional notes

`td-reorganize-doc` must be use in a machine where the `input_path` is available through the file system, i.e. not a machine which uses `ssh` to upload documents.

4 Generating HTML

4.1 How it works

`td-generate` will walk through the documentation tree and create HTML files for each directory. It looks for `.xml` files, which describe documents. If a document wants to be indexed, an associated `.xml` file have to be created. XML files can also describe directories, not only files.

`td-generate` will also look for `.dsc` files, and convert them into `.xml` files.

4.2 XML descriptions

Here is a sample XML description, corresponding to the previous `.dsc` file.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<document>
  <title>Fractals are everywhere in the world</title>
  <file>foobar.89.micai.fractals_are_everywhere_in_the_world.pdf</file>
  <nbpages>10</nbpages>
  <type>public</type>
  <author>Patrick Foobar</author>
  <date>1989</date>
  <language>english</language>
  <summary>Fractional Brownian motions number of physical phenomena<br>
fractional order. However, the precise meaning of such complementary<br>
approaches are The first one, based on nonstationary nature of<br>
measurements; the second one, self-similarity properties of FBM and<br>
reveals an underlying stationary structure relative to each<br>
time-scaling.</summary>
  <comment><content></content></comment>
  <url>http://patrick.foobar.com/fractals.pdf</url>
</document>
```

4.3 Using `td-generate`

Usage: `td-generate [options]`

Options

```
-c, --config configfile  Config file to use.
-v, --version            Print version number.
```

`td-generate` will read the configuration file, and then recursively creates HTML files. If you want `td-generate` not to go into a particular directory, create a `.td_raw_dir` file into it.

```
$ cd my_docs/mess
$ touch .td_raw_dir
```

This way `td-generate` will not consider this directory.

4.4 Configuration file

The default configuration file is commented, here is an example:

```
# Path where document, xml and dsc files are located.
input_path      = "/tmp/tidydoc"

# Path where generated files should be put.
output_path     = "/tmp/tidydoc"

# Template files path.
templates_path  = "/usr/local/share/tidydoc/templates"

# Root url of the website. Only useful with html.
site_root       = "file:///tmp/tidydoc"

## HTML parameters

# HTML files which have to be generated at the site root.
site_root_html_files = ["nav.html"]

# HTML files which have to be generated for each directory.
directories_html_files = ["index.html", "doclist.html"]

# Treeview file to link for each directory.
treeview_subdirs_link = "doclist.html"

# Whether treeview should be in multiframe mode or not.
treeview_multiframes = True
```

By using the default HTML template files, this will create documents with a navigation panel on the left frame, and the document descriptions on the right frame.

4.5 Template files

HTML output is driven by template files, which are processed by `td-generate`, by substituting some special variables into it. For example, the template file for `index.html` should be named `index.html.tpl`, and might look like this for a multiframe document:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN">

<html>
<head>
  <title>Documentation</title>
</head>

<frameset cols="35%,65%">
  <frame src="%%SITE_ROOT%%/nav.html" name="treeframe" id="treeframe">
  <frame src="doclist.html" name="basefrm" id="basefrm">
</frameset>
```


`</html>`

`%%SITE_ROOT%%` will be substituted by `td-generate`.

Some template files have a special meaning:

- `document.tpl` Processed to create a description for each document.
- `link.tpl` Processed to create a link for each document.
- `subdir.tpl` Processed to create a link for each subdirectory.

4.6 Using Treeview

In order to get a nice tree-like view, Treeview (<http://www.treeview.net>) is supported. To use it, you need to copy the whole directory `doc/examples/treeview` in your `output_path`. This way, `td-generate` will automatically create a `docNodes.js` linking your documents. You can also download directly the archive from the treeview web site.

5 Generating Bibtex

`td-generate` will generate a global bibtex file for all your documents. The output file can be specified in `tidydoc.conf`:

```
## Bibtex parameters

# Output .bib file.
bibtex_file = output_path + "/documents.bib"
```


