

Frugalware 1.5 (Nexon) Documentation

Contents

1	Security support	1
1.1	Introduction	1
1.2	Handling security bugs	1
1.3	How to release an FSA?	1
1.4	How to notice security issues	1
1.5	How to get patches	2
1.6	Versioning	2
2	Handling git repositories	2
2.1	Introduction	2
2.2	Name of the repository	2
2.3	Location of the repository	3
2.4	Registering for the gitweb interface	3
2.5	Enabling hooks for your repository	3
2.6	Setting up server configuration for a WIP repo	4
2.7	Enabling syncpkgd support for a WIP repo	4
3	This is a small tutorial for those who want to contribute to Frugalware	4
3.1	Ways of contributing	4
3.1.1	Translations (translators)	5
3.1.2	Necessary documentation (packagers, coders)	5
3.1.3	Downloading and setting up the repositories	5
	Getting the frugalware-current repo (packagers)	5
	Getting pacman-g2 and other code (coders)	6
	Setting up the repository and sending patch via email (packagers, coders)	6
3.1.4	Further options for those who have developer account (packagers, coders)	7
	Setting up the frugalware-* repos and repoman (packagers)	8
	Setting up other repos (coders)	8
4	GNOME Bump HOWTO	9
4.1	GNOME compile order	9
4.2	Bumping individual packages	14

5	Frugalware Release HOWTO	14
5.1	Introduction	14
5.2	A testing release	15
5.3	Preparing	15
5.4	Creating the stable tree	15
5.5	Updating the -current tree	15
5.6	Updating the -stable tree	16
5.7	Testing	16
5.8	Announcement	16
5.9	For the next release	16
6	Artwork requirements	16
6.1	Introduction	16
6.2	The rules	17
7	Table of user / group ids used in Frugalware	17
8	List of packages needs to be rebuilt after the given bumped	36
8.1	kernel	36
8.2	mysql	36
8.3	libgda	36
8.4	db	36
8.5	gnutls	37
8.6	dbus	37
8.7	dbus-mono	38
8.8	neon	38
8.9	binutils	38
8.10	libtasn1	38
8.11	gstreamer	38
8.12	gtk+2	39
8.13	libcdio	39
8.14	vte	39
8.15	firefox	40
8.16	xulrunner	40
8.17	wireless_tools	40
8.18	parted	40
8.19	libpqxx	40
8.20	openobex	40
8.21	bluez-libs	40
8.22	gail	41

8.23	imagemagick	41
8.24	evolution-data-server	41
8.25	x264	41
8.26	ocaml	41
8.27	openbox	41
8.28	pilot-link	41
8.29	php	41
8.30	libevent	42
8.31	exiv2	42
8.32	icu4c	42
8.33	c-ares	42
8.34	libofx	42
8.35	directfb	42
8.36	sword	43
8.37	gpm	43
9	Creating translations for init scripts	43
9.1	Preparing the source	43
9.2	Creating the pot file	43
9.3	Creating a po file	43
9.4	Creating the mo files	44
10	Frugalware AsciiDoc quickstart	44
10.1	Features	44
10.2	Restrictions	45
10.3	Skeleton for README.Frugalwares	45
10.4	Skeleton for standalone documentation	45
10.5	Buiding it on your own machine	46
10.6	Adding a new project to Pootle	47
11	Frequently Asked Developer Questions	47
11.1	What is the recommended way to version bump a package if I don't have git push access?	47
11.2	makepkg ends up with <packagename>: /usr/info/dir: exists in filesystem	47
11.3	I can't pacman-g2 -Su <package>, it says local version is newer, but I know it isn't!	47
11.4	What does 5.55 SBU mean?	47
11.5	Why do maintainers cry about my new package's tarball?	48
11.6	What should and shouldn't I include in depends(), rdepends() and makedepends()?	48
11.7	What are the various dependency-control arrays for?	48
11.8	How can I have PHP to work with my newly packaged eaccelerator/anything extension?	48
11.9	How can I cross-compile (package) an architecture-independent (non-binary) program?	49

11.10	repoman upd can't create /var/fst/ as it already exists	49
11.11	How can I access the central FW repo (mirrors are too slow for me)?	49
11.12	What should I write as patch name and long comment at repoman rec?	49
11.13	Where should I place my comments about a package?	49
11.14	I want to work with the latest development version of pacman&co.! How?	49
11.15	Naming locale packages	49
11.16	Error handling	50
11.17	Permissions	50
11.18	Stripping	50
11.19	When should I use \$Fsourcedir and \$Fdestdir	50
11.20	When should I increment a package's release number?	50
11.21	How do I repair a corrupted package database?	50
12	Frugalware Source Tree Testsuite	50
12.1	Introduction	50
12.2	Rules	51
12.3	Technical details	51
13	Translations	51
13.1	Introduction	51
13.2	Rules	51
13.3	Goals	52
13.4	Overview	52
14	How to port Frugalware to a new architecture	53
14.1	Introduction	53
14.2	Toolchain	53
14.3	Base system	53
14.4	The rest	53
15	GNU Free Documentation License	53
15.1	PREAMBLE	54
15.2	APPLICABILITY AND DEFINITIONS	54
15.3	VERBATIM COPYING	55
15.4	COPYING IN QUANTITY	55
15.5	MODIFICATIONS	55
15.6	COMBINING DOCUMENTS	56
15.7	COLLECTIONS OF DOCUMENTS	57
15.8	AGGREGATION WITH INDEPENDENT WORKS	57
15.9	TRANSLATION	57
15.10	TERMINATION	57
15.11	FUTURE REVISIONS OF THIS LICENSE	57

Copyright (C) 2005, 2006, 2007, 2008, 2009, 2010, 2011 The Frugalware Developer Team.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

1 Security support

1.1 Introduction

This document documents the work of the Frugalware Security Team. Primarily it's for new developers or for existing developers who join the Security Team.

1.2 Handling security bugs

1. The security team opens a new task in the BTS, with a [SEC] prefix.
2. The maintainer fixes the issue in -current and decides if the issue needs fixing in -stable or not. If yes, then changes the status of the task to "Fixed in -current", otherwise closes the task.
3. If there is no patch for the issue yet, then set the status to "Researching". This indicates that you, the maintainer, is aware of the problem, but don't yet have enough a solution.
4. The security team regularly searches for "Fixed in -current" bugs, fixes the issue in -stable and releases a new FSA.

1.3 How to release an FSA?

1. Check if the backport built by syncpkgd is ready (the binary packages should be uploaded for each arch).
2. Open the -stable Changelog file of the package. There you can see the vulnerable and unaffected versions of the package.
3. Add a new entry to the frugalware/xml/security.xml file in the homepage-ng repo.
4. Commit, push. The commit hook will check if the xml is valid, so most common errors can be avoided. In rare cases, the announcement may not appear on the frugalware-security list. If this is the case, then ask on -devel about what the problem might be.
5. Close the task in the BTS, filing in FSAxxx in the closure message.

1.4 How to notice security issues

1. Subscribe to Secunia Security Advisories List at http://secunia.com/secunia_security_advisories/ page. This is the best place to notice issues.
2. Read the mails one-by-one and check if the advisory affects -current or -stable.
3. Open a task in BTS if necessary. Please fill in the form correctly, provide a patch if you can.

You can also read other mailing lists, like <https://lists.grok.org.uk/mailman/listinfo/full-disclosure>, but Secunia monitors them, so you won't miss anything. (You just notice things later.)

1.5 How to get patches

Secunia announces security issues days after they released so there is a good chance to find a patch.

1. First of all sometimes upstream fixes it with a new version.
2. Fixed in cvs/svn/whatever and you are able to find the patch (unlike PHP)
3. If these two fail, there is <http://security.ubuntu.com/ubuntu/pool>. Secunia also mails you if the bug fixen in Ubuntu, so steal the patch from them :) You only need the \$package-\$pkgver.diff.gz. There is a changelog in it, where you can find the filename of the fix.
4. It's also a good idea to take a look on RedHat/Gentoo bugzilla. They attach fixes most the time.

So it's good to read the Secunia mails carefully as you'll always know when the patch is available.

1.6 Versioning

We use integers in pkgrels for normal packages, but -stable updates are different. Here are the cases:

- If you do a version bump (we refer to them as *secfix bump* usually in -stable commit messages), then you need to set pkgrel to 1<release_codename>1.
- If you add a security patch, and pkgrel was an integer (let's say *I*), then you should increment pkgrel to 2<release_codename>1.
- If you add a security patch when the pkgrel was already in an X<release_codename>Y form, increment it to X<release_codename>Y+1 (Alternatively, you can use X+1<release_codename>Y if there is already a newer version in -current.)

This ensures that:

- The version of the security update will be larger than the one in -stable, so that the package will be upgraded when the user does a pacman-g2 -Syu on -stable.
- The version of the security update will be smaller than the one in -current, so that the package will be upgraded when the user upgrades to a new version (current or new stable).

2 Handling git repositories

2.1 Introduction

This document is for developers who want to publish a git repository on the Frugalware FTP Server and on the Frugalware Gitweb Interface.

2.2 Name of the repository

The name of the official repositories are `frugalware-current`, `frugalware-stable` and so on.

The name of WIP repositories are typically in a `featureNUM` form, like `kde45` or `parted2`, referring to the name of the software it contains and its version. This method is used so that the repository name can be a valid shell variable as well.

Please note that there is a convention that WIP repository names never contain a hyphen (-). This is on purpose. It's not trivial to decide that when you merge code from one repository to another then build servers should try to build automatically the new packages you brought in or not. Because of this the policy is that if a hyphen is in the name, the it'll build the new packages (WIP → -current merge), but it won't do so when you merge the other way around.

2.3 Location of the repository

Since a repository consists of plain files, we can and should place them on the ftp server (/home/ftp). To prevent further problems, always use the server name "git.frugalware.org", currently it's an alias of genesis.frugalware.org.

First decide if it's a personal repository or a team one. For example if you create a repository to update to a newer python version, then you will probably do all the work, create it under /pub/other/people/nick/reponame. Simply create a dir, issue `git init` and push at least one commit to there (but before pushing, enable the hooks, see below).

Now anyone can `git clone` it, using a *full mirror*, for example `ftp://ftp12.frugalware.org/mirrors/ftp.frugalwa`

2.4 Registering for the gitweb interface

If the repository is a team one, then create it under /pub/other. In this case you probably want the gitweb interface, too. To use it:

1. Update the file `.git/description` inside the repo with a short (less than 80 chars) description.
2. Create the file `.git/owner` inside the repo containing your name, *without* your email address.
3. Push a *relative* symlink to the homepage-ng repository, see the existing ones as a reference.

After some time (a maximum of 30 minutes) it should appear at `http://git.frugalware.org/`.

2.5 Enabling hooks for your repository

Currently you need hooks for the following reasons: . If you don't use *bare* repositories, then the content outside `.git` won't be updated automatically, you need a hook to do so.

1. If you want CIA notification.
2. If you want to send mails to the Frugalware-git mailing list.
3. If you want to let others clone your repository via *dumb* protocols like `http` or `rsync`. (This means that if you disable this hook, it won't be accessible anonymously!)

For the last one:

```
mv .git/hooks/post-update{.sample,}
echo "unset GIT_DIR; cd ..; git checkout -f" > .git/hooks/post-receive
chmod +x .git/hooks/post-receive
```

For the others:

```
ln -sf /home/ftp/pub/other/git-hooks/git-hooks.py .git/hooks/post-receive
```

One thing that a hook won't do for you is to allow pushing to the master branch, even if it's the checked out one. This is normally not good, but our hook will handle this, so we can ignore the problem:

```
git config receive.denyCurrentBranch ignore
```

2.6 Setting up server configuration for a WIP repo

When you run `repoman`, it invokes `repoman server` on the remote machine using `ssh`. `repoman server`, just like plain `repoman`, reads configuration from `/etc/repoman.conf` and `$HOME/.repoman.conf`, so you need to set up the later before you can push packages to your WIP repo.

Here is a minimal example:

```
fst_root=/home/nick/git
repos=('current' 'mywiprepo')
```

And then you have to symlink the repos to `$HOME/git`, for example:

```
cd $HOME/git
ln -s /pub/frugalware/frugalware-current current
ln -s /pub/other/people/nick/nicktesting/ nicktesting
```

2.7 Enabling `syncpkgd` support for a WIP repo

If you create a new WIP repo, `syncpkgd` won't sync packages in it by default.

This means that if you just push your commits, no attempt will be made to build the relevant binary package automatically for you, which is the case for the `-current` / `-stable` repos.

If you want `syncpkgd` support, then you need to edit 3 configuration files on the server which runs `syncpkgd` (that's typically not your local machine and not the one that runs `syncpkgcd`).

Edit `syncpkgd`'s `repoman` config by extending the `repos` array and adding the `foo_servers` and `foo_sudo` variables:

```
vi ~syncpkgd/.repoman.conf
```

Add a `pacman-g2` configuration file:

```
vi ~syncpkgd/.pacman-g2/repos/foo
```

The contents will be something like this:

```
[foo]
Server = http://ftp.frugalware.org/pub/other/people/nick/foo/frugalware-@CARCH@
```

Note

Don't replace `@CARCH@` with anything else, `syncpkgcd` will do so later!

Finally edit the `git` hook and add `foo` to the end of the `repos` array:

```
vi /pub/other/git-hooks/synchook/config.py
```

If you no longer need these entries, you can remove them, but leave at least one there as an example.

3 This is a small tutorial for those who want to contribute to Frugalware

3.1 Ways of contributing

There are many different ways to contribute to Frugalware. You can write documentation, translate the existing documentation into your native language (or any other language you want to), maintain packages or improve them with added features etc.

If you are a programmer you can help us in developing our applications. These are: pacman-g2, gfpmp, fwlive, frugalwareutils, setup etc. See git.frugalware.org for different project repositories.

You can also start new projects. If you show some code we can surely host your project too if it's Frugalware related. For example you want to write kfpm :)



Important

After each title in brackets you can find the target audience.

3.1.1 Translations (translators)

You can read the details on our [Translations](#) documentation page.

3.1.2 Necessary documentation (packagers, coders)

In the first part I will cover the information necessary for those who do *not* have developer status yet.

In the second part we will set up the necessary config files.

First of all, we ask you to read the following documentation carefully. If you do not want to deal with packages, but just want to code it's usually enough to read the git documentation as we store our code in git repositories.

- man makepkg
- man pacman-g2
- man repoman
- man FrugalBuild
- man fwmakepkg
- [makepkg howto](#)
- [git getting started](#)

I know, it is boring reading documentation, but you have to know that writing it is even worse so do not ask questions when the answer in the documentation. If you can not understand something feel free to join [#frugalware@irc.freenode.net](https://irc.freenode.net) and ask.

3.1.3 Downloading and setting up the repositories

Getting the frugalware-current repo (packagers)

The frugalware-current repo is the development repo for the packages.

When you want to get it you need the git package. Let's get it:

```
# pacman-g2 -S git
```

Now create a git directory where you can hold all your repos. You can choose any other name of course.

```
$ mkdir -p ~/git  
$ cd ~/git
```

Now clone the repo with git:

```
$ git clone http://frugalware.org/git/pub/frugalware/frugalware-current current  
$ cd current
```

Now be patient while git clones all the objects and then checks out the files. Also you can use other mirrors as well.

Getting pacman-g2 and other code (coders)

First of all you need the repo of the program. In this example I will use pacman-g2, but the steps are very similar. NOTE: Most of our programs need the translations repo to compile)

```
$ mkdir -p ~/git
$ cd ~/git
$ git clone http://frugalware.org/git/pub/other/translations (optional)
$ git clone http://frugalware.org/git/pub/other/pacman-g2/pacman-g2
$ cd pacman-g2
```

Setting up the repository and sending patch via email (packagers, coders)

Now you should setup up your identity.

```
$ git config --global user.name "Your Name"
$ git config --global user.email email@addr.ess
$ git config branch.master.rebase true
```

Now you can make your changes. When finished run

```
$ git diff
```

in the repository.

Tip

You can also use *git diff .* (note the dot in the end). In that case git will show the changes recursively in the current directory. It is very handy when you have lot of uncommitted changes in your repo.

If you are satisfied with the changes run

```
$ git commit -a
```

to commit all your changes.

If you want to cherry-pick hunks from your changes:

```
$ dg record
```

or using native git commands:

```
$ git add -p; git commit
```

Without committing your changes you can not send nor push (just developers) it.

Tip

With frugalware-* repos it's recommended to use *repoman rec* which is a wrapper for dg record. It sets up the patch name properly so you only need to deal with the details.

Note

[Here](#) you can find more details on how to write good commit messages in general using git.

Here comes the final step. Send in the patch(es)!

```
$ git format-patch <hash>
$ git send-email --to frugalware-devel@frugalware.org .
```

<hash> is the sha1 of the last patch you do *not* want to submit. Run

```
$ git log
```

and you'll see the hash. Also, you can just use your existing mail client and send the patch(es) as an attachment.

If everything goes fine your patch should show up on the [frugalware-devel](#) mailing list soon.

Note

You have to subscribe to the [frugalware-devel](#) mailing list and set up your SMTP server properly (if you use `git send-email`).

It doesn't really belong to here but I want to document it somewhere. If you are a developer and want to apply such a patch, you need:

- Check the patch itself. If the second line is not an empty one, then you need to hand-edit the patch before applying:

```
Subject: [PATCH] powwow-1.2.13-1-i686
* new package
```

to:

```
Subject: [PATCH] powwow-1.2.13-1-i686
* new package
```

- Then you can apply the patch using `git-am`:

```
$ cat 0002-powwow-1.2.13-1-i686.patch | git am
```

You should do this in the root directory of the repository.

3.1.4 Further options for those who have developer account (packagers, coders)

Once you get a developer account, you have the right to request the following services:

- BTS access (so that we can assign tasks to you)
- git write access (you'll always get this, except if you are working on the artwork or so)
- voice on the [#frugalware.dev](#) channel
- a [@frugalware.org](#) mail address (with `imaps/pop3s` access)
- Public and private devspace. The first is in the `/pub/other/people/nick` dir and this is mirrored (you must not put private stuff to there). The later is your `~/public_html` dir: it is not mirrored and there is no backup for it. Though you may temporarily put private stuff to there.
- a [@frugalware.org](#) jabber account if you want one

What you should do:

- You should read the frugalware-devel mailing list. When you're asked, please try to respond.
- If you push patches to git, you should subscribe to the frugalware-git mailing list. This list has a big traffic since a new mail is sent for each patch. If you don't have time to read it, subscribe then set the "I would like to receive no mail" option. Also take care that your subscribing email address is the same one you set using `git config user.email`
- It's good if you can join the user and developer channel when you're online.
- Maintain your packages. Try to resolve your assigned bugs, try to keep your packages up to date, and if you needed patches for packages, send them upstream. If you don't have anything to do for a week that's usually a bad sign. It's - of course - OK when you go for vacation a few times a year, but then please announce it on the developer mailing list so that we won't wait for you when fixing urgent problems, etc.
- Document your work. The documentation is worth nothing if it's outdated. Ideally someone who has never contacted us should be able to understand every detail of Frugalware, just from documentation. No secrets! We are not kids.
- If you have time, try to read the mailing lists (`frugalware-users*`) and the forums. If you prefer reading the forums from your mail client, there is a bi-directional gateway on the `frugalware-forums@` list, use it.

Let us see what you should set up to get it work. I will also give some tips which can make your life easier.

Read [this page](#), we collected a set of tricks when we converted from darcs to git.

Setting up the frugalware-* repos and repoman (packagers)

It is time to set up some necessary things. We start with the frugalware-current repo. Make sure that you are in the root of the frugalware-current repo. Also do not forget to change the username to your login name on `git.frugalware.org`.

```
$ git config remote.origin.url 'username@git.frugalware.org:/home/ftp/pub/frugalware/ ↵  
  frugalware-current'  
$ git config remote.origin.receivepack "sudo -u repo git-receive-pack"
```

As you will use `repoman` to upload the packages (and many other things as you'll see) we should set it up now. This step is also necessary. Open `~/repoman.conf` with your favourite editor and add the following lines:

```
fst_root=~/.git  
current_servers=("username@git.frugalware.org:/home/ftp/pub/frugalware/frugalware-current")  
stable_servers=("username@git.frugalware.org:/home/ftp/pub/frugalware/frugalware-stable")  
stable_pushonly="y"
```

Where `fst_root` is the directory where you store your git repos. Username is your login on `git.frugalware.org`. For details see `man repoman`.

As from now use the following command from package's directory to push your changes.

```
$ repoman push
```

It will check the FrugalBuild using `fbint`, then record your changes, push them, upload the `fpms` and finally create the changelog, update the `fdb` etc. So you are done if there was no error message.

Setting up other repos (coders)

In repo's main directory:

```
$ git config remote.origin.url 'username@git.frugalware.org:/home/ftp/pub/other/pacman-g2/ ↵  
  pacman-g2'  
$ git config remote.origin.receivepack "sudo -u owner git-receive-pack"
```

Do not forget to change the username and repository path. For paths refer to the [gitweb](#) interface.

Note

The owner for `pacman-g2`, `frugalwareutils`, `pacman-tools` is usually `vmiklos`.

You should always review what you would push before you perform the action:

```
$ git fetch
$ git rebase origin/master
$ git log origin/master..master
```

Then you can use

```
$ git push
```

to send in your changes.

Note

The `dg push` wrapper does exactly this for you.

4 GNOME Bump HOWTO

You **MUST** follow this HOWTO when bumping GNOME to a new version (even a minor version).

To start, packages must be compiled in the order listed below (if you find a change that needs to be made to this list, poke Bouleetbil). If it is a major bump (2.14 to 2.16, for example), it is wise to rebuild most of the GNOME packages.

4.1 GNOME compile order

- `libxml2`
 - `libxslt`
 - `gnome-common`
 - `intltool`
 - `rarian`
 - `gtk-doc`
 - `glib`
 - `libIDL`
 - `ORBit2`
 - `libbonobo`
 - `fontconfig`
 - `Render`
 - `Xrender`
 - `cairo`
 - `cairomm`
 - `Xft`
-

- pango
 - atk
 - shared-mime-info
 - gtk*
 - gtk+2-engines
 - gtkmm
 - gconf
 - desktop-file-utils
 - gnome-mime-data
 - avahi
 - avahi-glib
 - dbus
 - hal
 - gamin
 - dbus-glib
 - libgnome-keyring
 - gnome-keyring
 - libproxy
 - libsoup
 - gvfs
 - gnome-vfs
 - audiofile
 - esd
 - libgnome
 - libart_lgpl
 - libglade
 - libgnomecanvas
 - libbonoboui
 - hicolor-icon-theme
 - icon-naming-utils
 - gnome-icon-theme
 - libgnomeui
 - startup-notification
 - gnome-themes
 - gnome-doc-utils
-

- gnome-desktop
 - libwnck
 - libgpg-error
 - libgcrypt
 - libtasn1
 - opencdk
 - gnutls
 - firefox
 - libgweather
 - evolution-data-server
 - pygobject (*)
 - pycairo
 - pygtk (*)
 - gnome-menus
 - librsvg
 - libcanberra-gtk
 - gnome-panel
 - zenity
 - metacity
 - gstreamer
 - liboil
 - libxklavier
 - libgnomekbd
 - libcroco
 - eel
 - gst-plugins-base
 - gnome-settings-daemon
 - nautilus
 - control-center
 - gnome-session
 - vt
 - gnome-terminal
 - libgtop
 - gucharmap
 - gnome-applets
-

- libgsf
 - libgnomecup
 - libgnomeprint
 - libgnomeprintui
 - yelp
 - bug-buddy
 - gtksourceview
 - pygtksourceview
 - pyorbit (*)
 - gnome-python (*)
 - iso-codes
 - totem-pl-parser
 - totem
 - brasero
 - gnome-media
 - eog
 - poppler
 - evince
 - gedit
 - gnome-python-desktop
 - alacarte
 - nautilus-cd-burner
 - gst-plugins-good
 - libmusicbrainz
 - gconf-editor
 - gnome-utils
 - gnome-system-monitor
 - gnome-netstatus
 - gcalctool
 - at-spi
 - libgail-gnome
 - gnome-speech
 - gnome-mag
 - gnopernicus (missing from repo)
 - gok (missing from repo)
-

- epiphany
 - epiphany-extensions
 - gob2
 - gnome-games
 - gnome-user-docs
 - file-roller
 - gnome-nettool
 - vino
 - vinagre
 - gnome-volume-manager
 - gnome-backgrounds
 - sound-juicer
 - gtkhtml
 - gal
 - pilot-link (if needed, not a gnome part)
 - gnome-pilot
 - gnome-pilot-conduits
 - gnome-spell
 - evolution
 - evolution-webcal
 - evolution-exchange
 - gdm
 - ptlib
 - opal
 - ekiga
 - dasher
 - gnome-power-manager
 - gnome-keyring-manager
 - deskbar-applet
 - fast-user-switch-applet
 - gnome-screensaver
 - pessulus
 - sabayon
 - gnome-cups-manager
 - system-tools-backends
-

- liboobs
- cheese
- gnome-system-tools
- mousetweaks
- seahorse
- gnome-sharp
- gnome-desktop-sharp
- empathy
- hamster-applet
- nautilus-sendto

(*) - don't use Fsplit on this package.

Note

all *sharp and all bindings need to be split

4.2 Bumping individual packages

Never, I repeat, **NEVER** bump a version without doing the following:

1. Download the new version's tarball and extract it
2. Run `./configure --help` and look in `configure.in` to check for new dependencies (even optional ones) and consider whether to use them or not. Consult all devels about whether it is a good idea to use the optional dependencies.
3. Check for dependencies that are no longer needed and remove them from the FrugalBuild
4. Check GConf schemas. Sometimes they have been renamed, or new ones have been added. Not doing this can cause a lot of problems.
5. Check the Changelog and NEWS file for the package. Sometimes there may be API/ABI changes that need to be considered before bumping.
6. Check if `_F_gnome_{scrollkeeper,mime,desktop}` are needed in the new version.
7. When all this has been done, update the FrugalBuild with new sha1sums, pkgver, depends, GConf schemas and `_F_gnome_*` values (add `gnome-scriptlet` to `Finclude` if necessary)
8. Build the package and push.

5 Frugalware Release HOWTO

5.1 Introduction

The aim of this howto is to show what's the procedure of a stable Frugalware release. The to-be-created release in this howto is 0.5, the previous release is 0.4.

5.2 A testing release

A testing release is similar to a full one, but much simpler. Here are the steps:

- bump the `frugalware` package: update the `Makefile` in `frugalware.git`, upload a new release tarball, and update the package in `-current`
- rebuild the `setup` package, update the version of the `frugalware` package dependency to the new version
- wait for the nightly cronjob to publish `setup kernel+initrd` under `/pub/frugalware/frugalware-current/boot`
- now you can generate a netinstall iso using `mkiso` for a single architecture you can test and upload the image to `/pub/frugalware/`
- do a default install and make sure the machine boots up and you can log in using the graphical interface (if not, then fix it)
- run `dg tag <version>` for the new version and push it
- sync changes from `-current` to `-testing`:

```
$ rsync -avP --delete-after frugalware-current/ frugalware-testing/
```

- generate installer images for a single architecture using `mkisorelease`
- wait at least 24h so that mirrors will be in sync
- update `news.xml` and `roadmap.xml` to mark the release as done

5.3 Preparing

- send a mail to `-devel` about "please stop version and release bumps"
- check if the artwork has been updated completely. see [this](#) mail from Nadfoka on what items should be checked
- ask someone to update the screenshots
- sync the archs, `checkpkgs` shouldn't have any red pkg in it's output
- run `gensync` to rebuild the `fdbs`
- generate isos and test if everything is ok (ie. install from `cd1-cd2` on `i686`, and start `kde`, or something)
- check if the upgrade from `0.4`→`0.5` works or not, probably a simple `-Syu` is not enough, then write a howto
- tag the release using `git tag`

5.4 Creating the stable tree

Copy the full tree on genesis:

```
$ cd /home/ftp/pub/frugalware  
$ cp -av frugalware-current frugalware-0.5
```

5.5 Updating the `-current` tree

Now one has two trees. All what one should do in `-current` is to regenerate `ChangeLog.txt` (copy & paste the command from `tools/genpkgdbs`).

5.6 Updating the -stable tree

- rename the frugalware-current fdfs to frugalware
- run tools/mkpkgst for each arch
- update VERSION in docs/Makefile, and rebuild the manual
- update `\.git/description`
- run genpkgs to regenerate the ChangeLog.txt to start from the 0.4 tag to the 0.5 tag
- update pacman-{g2,-tools} and fwsetup so that -stable will be the default on -Syu / repoman upd / in the installer, not -current
- upload the fdfs to the mysql db using fpm2db, just run all2db.sh from the /tools dir
- create a new chroot tarball for each arch

5.7 Testing

- generate isos, test *all* of them (net,cd,dvd for each arch)
- create an usb stick installer tarball for each arch
- create an tftp boot image for each arch
- create a gui installer image for each arch

5.8 Announcement

- put the isos online and wait at least 24h so that the mirrors will be in sync at release time
- create torrents for the isos and make sure at least one machine seeds them
- add the new version to the bts
- write an announcement, put it out to somewhere and ask Alex or LGee to spellcheck it
- push it to the homepage-ng repo
- mark the release as "done" in `/frugalware/xml/roadmap.xml` (homepage-ng repo) and add the proper newsid value
- update the topic of #frugalware
- update the freshmeat entry

5.9 For the next release

- find a codename
- update roadmap.xml

Done!

6 Artwork requirements

6.1 Introduction

This document details the requirements that must be met by all artwork if it is to be accepted into the official Frugalware gallery.

6.2 The rules

- All artwork must be licensed under the Free Art License 1.3 ([full details](#)).
- Where the Frugalware logo appears, only the officially approved logo may be used. Refer [here](#) for the logo.

Note

There is a newer SVG version available [here](#).

- Artwork must be submitted in either SVG or XCF (The Gimp) format as this allows for derivative works to be made without affecting the impact of the original artwork. Examples of derivative works include wallpapers in various sizes and height/width ratios, and/or KDM/GDM/SLiM themes. To suit the varying sizes and ratios of monitors, any wallpaper must be a minimum 1600 pixels wide and provided in both 4:3 and 16:9 ratios.
- All artwork must be submitted together with any associated source files - i.e. files which are required by the graphics editor used by the entrant to reproduce and/or edit the artwork.
- Only FLOSS software may be used to create the wallpaper.
- Neither the release's version number, nor code-name are to appear in artwork, or there should be a version without them for later use when a given release is no longer supported.

7 Table of user / group ids used in Frugalware

Table 1: Users and groups that are added with a specific uid/gid

ID	User	Package	Group	Package
000	root	shadow	root	shadow
001	bin	shadow	bin	shadow
002	daemon	shadow	daemon	shadow
003	adm	shadow	sys	shadow
004	lp	shadow	adm	shadow
005	sync	shadow	tty	shadow
006	shutdown	shadow	disk	shadow
007	halt	shadow	lp	shadow
008	mail	shadow	mem	shadow
009	news	shadow	kmem	shadow
010	uucp	shadow	wheel	shadow
011	operator	shadow	floppy	shadow
012	syncpkgd	pacman-tools	mail	shadow
013			news	shadow
014	ftp	shadow	uucp	shadow
015			man	shadow
016			cdrom	shadow
017			scanner	shadow
018	privoxy	privoxy	privoxy	privoxy
019	fst	pacman	audio	shadow
020	nx	freenx	games	shadow
021			slocate	slocate
022			utmp	shadow
023			camera	shadow
024			video	shadow
025	smmsp	shadow	smmsp	shadow
026	clamav	clamav	clamav	clamav

Table 1: (continued)

ID	User	Package	Group	Package
027	mysql	shadow	mysql	shadow
028	rsyncd	rsync	rsyncd	rsync
029	_ntp	openntpd	_ntp	openntpd
030			storage	shadow
031	pgdb	postgresql	pgdb	postgresql
032	rpc	shadow	rpc	shadow
033	sshd	shadow	sshd	shadow
034	scponly	scponly	scponly	scponly
035			sbox	scratchbox
036			rlocate	rlocate
037			netdev	shadow
038	messagebus	dbus	messagebus	dbus
039	hald	hal	hald	hal
040	amavis	amavisd-new	amavis	amavisd-new
041	ejabberd	ejabberd	ejabberd	ejabberd
042	gdm	shadow	gdm	shadow
043			shadow	shadow
044	beagleindex	beagle	beagleindex	beagle
045	partimag	partimage	partimag	partimage
046	sabayon	sabayon	sabayon	sabayon
047	munin	munin and munin-node	munin	munin and munin-node
048			ccache	ccache
049	openldap	openldap	openldap	openldap
050			ftp	shadow
051			telnetd	shadow
052			tape	shadow
053			dialout	shadow
054	prosody	prosody	prosody	prosody
055			lock	systemd
056				
057			realtime	pulseaudio
058			pulse-access	pulseaudio
059	pulse	pulseaudio	pulse	pulseaudio
060			grsec_procview	kernel-grsec
061			grsec_audit	kernel-grsec
062			grsec_tpe	kernel-grsec
063			grsec_s_all	kernel-grsec
064			grsec_s_client	kernel-grsec
065			grsec_s_server	kernel-grsec
066	mediatomb	mediatomb	mediatomb	mediatomb
067	polkituser	policykit	polkituser	policykit
068	usbmuxd	usbmuxd	usbmuxd	usbmuxd
069	couchdb	couchdb	couchdb	couchdb
070				
071				
072				
073	postfix	postfix	postfix	postfix
074				
075			postdrop	postfix
076				
077	dspam	dspam	dspam	dspam
078				
079				

Table 1: (continued)

ID	User	Package	Group	Package
080	mailman	mailman	mailman	mailman
081				
082	exim	exim	exim	exim
083				
084	avahi	avahi	avahi	avahi
085	firebird	firebird	firebird	firebird
086				
087				
088				
089				
090	pop	shadow	pop	shadow
091				
092				
093				
094				
095				
096				
097				
098			nobody	shadow
099	nobody	shadow	nogroups	shadow
100			users	shadow
101		shadow	console	shadow
102				
103				
104	distccd	distcc	distccd	distcc
105				
106				
107				
108				
109	postgrey	postgrey		
110				
111				
112				
113	logcheck	logcheck	logcheck	logcheck
114				
115				
116				
117				
118				
119				
120				
121				
122				
123				
124				
125				
126				
127				
128				
129				
130				
131				
132				
133				

Table 1: (continued)

ID	User	Package	Group	Package
134				
135				
136				
137				
138				
139				
140				
141				
142				
143				
144				
145				
146				
147				
148				
149				
150	quagga	quagga	quagga	quagga
151				
152				
153				
154				
155				
156				
157				
158				
159				
160				
161				
162				
163				
164				
165				
166				
167				
168				
169				
170				
171				
172				
173				
174				
175				
176				
177				
178				
179				
180				
181				
182				
183				
184				
185				
186				
187				

Table 1: (continued)

ID	User	Package	Group	Package
188				
189				
190				
191				
192				
193				
194				
195				
196				
197				
198				
199				
200				
201				
202				
203				
204				
205				
206				
207				
208				
209				
210				
211				
212				
213				
214				
215				
216				
217				
218				
219				
220				
221				
222				
223				
224				
225				
226				
227				
228				
229				
230				
231				
232				
233				
234				
235				
236				
237				
238				
239				
240				
241				

Table 1: (continued)

ID	User	Package	Group	Package
242				
243				
244				
245				
246				
247				
248				
249				
250				
251				
252				
253				
254				
255				
256				
257				
258				
259				
260				
261				
262				
263				
264				
265				
266				
267				
268				
269				
270				
271				
272				
273				
274				
275				
276				
277				
278				
279				
280				
281				
282				
283				
284				
285				
286				
287				
288				
289				
290				
291				
292				
293				
294				
295				

Table 1: (continued)

ID	User	Package	Group	Package
296				
297				
298				
299				
300			jupiter	jupiter
301				
302				
303				
304				
305				
306				
307				
308				
309				
310				
311				
312				
313				
314				
315				
316				
317				
318				
319				
320				
321				
322				
323				
324				
325				
326				
327				
328				
329				
330				
331				
332				
333				
334				
335				
336				
337				
338				
339				
340				
341				
342				
343				
344				
345				
346				
347				
348				
349				

Table 1: (continued)

ID	User	Package	Group	Package
350				
351				
352				
353				
354				
355				
356				
357				
358				
359				
360				
361				
362				
363				
364				
365				
366				
367				
368				
369				
370				
371				
372				
373				
374				
375				
376				
377				
378				
379				
380				
381				
382				
383				
384				
385				
386				
387				
388				
389				
390				
391				
392				
393				
394				
395				
396				
397				
398				
399				
400				
401				
402				
403				

Table 1: (continued)

ID	User	Package	Group	Package
404				
405				
406				
407				
408				
409				
410				
411				
412				
413				
414				
415				
416				
417				
418				
419				
420				
421				
422				
423				
424				
425				
426				
427				
428				
429				
430				
431				
432				
433				
434				
435				
436				
437				
438				
439				
440				
441				
442				
443				
444				
445				
446				
447				
448				
449				
450				
451				
452				
453				
454				
455				
456				
457				

Table 1: (continued)

ID	User	Package	Group	Package
458				
459				
460				
461				
462				
463				
464				
465				
466				
467				
468				
469				
470				
471				
472				
473				
474				
475				
476				
477				
478				
479				
480				
481				
482				
483				
484				
485				
486				
487				
488				
489				
490				
491				
492				
493				
494				
495				
496				
497				
498				
499				
500				
501				
502				
503	bitlbee	bitlbee	bitlbee	bitlbee
504				
505				
506				
507				
508				
509				
510				
511				

Table 1: (continued)

ID	User	Package	Group	Package
512				
513				
514				
515				
516				
517				
518				
519				
520				
521				
522				
523				
524				
525				
526				
527				
528				
529				
530				
531				
532				
533				
534				
535				
536				
537				
538				
539				
540				
541				
542				
543				
544				
545				
546				
547				
548				
549				
550				
551				
552				
553				
554				
555				
556				
557				
558				
559				
560				
561				
562				
563				
564				
565				

Table 1: (continued)

ID	User	Package	Group	Package
566				
567				
568				
569				
570				
571				
572				
573				
574				
575				
576				
577				
578				
579				
580				
581				
582				
583				
584				
585				
586				
587				
588				
589				
590				
591				
592				
593				
594				
595				
596				
597				
598				
599				
600				
601				
602				
603				
604				
605				
606				
607				
608				
609				
610				
611				
612				
613				
614				
615				
616				
617				
618				
619				

Table 1: (continued)

ID	User	Package	Group	Package
620				
621				
622				
623				
624				
625				
626				
627				
628				
629				
630				
631				
632				
633				
634				
635				
636				
637				
638				
639				
640				
641				
642				
643				
644				
645				
646				
647				
648				
649				
650				
651				
652				
653				
654				
655				
656				
657				
658				
659				
660				
661				
662				
663				
664				
665				
666				
667				
668				
669				
670				
671				
672				
673				

Table 1: (continued)

ID	User	Package	Group	Package
674				
675				
676				
677				
678				
679				
680				
681				
682				
683				
684				
685				
686				
687				
688				
689				
690				
691				
692				
693				
694				
695				
696				
697				
698				
699				
700				
701				
702				
703				
704				
705				
706				
707				
708				
709				
710				
711				
712				
713				
714				
715				
716				
717				
718				
719				
720				
721				
722				
723				
724				
725				
726				
727				

Table 1: (continued)

ID	User	Package	Group	Package
728				
729				
730				
731				
732				
733				
734				
735				
736				
737				
738				
739				
740				
741				
742				
743				
744				
745				
746				
747				
748				
749				
750				
751				
752				
753				
754				
755				
756				
757				
758				
759				
760				
761				
762				
763				
764				
765				
766				
767				
768				
769				
770				
771				
772				
773				
774				
775				
776				
777				
778				
779				
780				
781				

Table 1: (continued)

ID	User	Package	Group	Package
782				
783				
784				
785				
786				
787				
788				
789				
790				
791				
792				
793				
794				
795				
796				
797				
798				
799				
800				
801				
802				
803				
804				
805				
806				
807				
808				
809				
810				
811				
812				
813				
814				
815				
816				
817				
818				
819				
820				
821				
822				
823				
824				
825				
826				
827				
828				
829				
830				
831				
832				
833				
834				
835				

Table 1: (continued)

ID	User	Package	Group	Package
836				
837				
838				
839				
840				
841				
842				
843				
844				
845				
846				
847				
848				
849				
850				
851				
852				
853				
854				
855				
856				
857				
858				
859				
860				
861				
862				
863				
864				
865				
866				
867				
868				
869				
870				
871				
872				
873				
874				
875				
876				
877				
878				
879				
880				
881				
882				
883				
884				
885				
886				
887				
888				
889				

Table 1: (continued)

ID	User	Package	Group	Package
890				
891				
892				
893				
894				
895				
896				
897				
898				
899				
900				
901				
902				
903				
904				
905				
906				
907				
908				
909				
910				
911				
912				
913				
914				
915				
916				
917				
918				
919				
920				
921				
922				
923				
924				
925				
926				
927				
928				
929				
930				
931				
932				
933				
934				
935				
936				
937				
938				
939				
940				
941				
942				
943				

Table 1: (continued)

ID	User	Package	Group	Package
944				
945				
946				
947				
948				
949				
950				
951				
952				
953				
954				
955				
956				
957				
958				
959				
960				
961				
962				
963				
964				
965				
966				
967				
968				
969				
970				
971				
972				
973				
974				
975				
976				
977				
978				
979				
980				
981				
982				
983				
984				
985				
986				
987				
988				
989				
990				
991				
992				
993				
994				
995				
996				
997				

Table 1: (continued)

ID	User	Package	Group	Package
998				
999				

8 List of packages needs to be rebuilt after the given bumped

8.1 kernel

For current:

```
revdep-rebuild 276
```

If you want syncpkgd to do the job:

```
revdep-rebuild 276 --nobuild --nopush
```

Note

Please use this only on minor (ie. 2.6.22.1 → 2.6.22.2) bumps, on a major bump many packages need fixing manually.

For solaria:

```
revdep-rebuild 41222 -t stable --nobuild --nopush
```

8.2 mysql

Only in case sover increases, for example if you update to 5.5.10:

```
git grep 'depends.*libmysqlclient>=' |grep -v 5.5.10
```

8.3 libgda

(maybe need rebuild)

- gnumeric
- libgnomedb

8.4 db

(only on major bumps, ie. 4.2.x → 4.3.x)

```
$ git grep "'db>="
```

about 28 packages at the moment.

8.5 gnutls

- bitlbee (.so)
- claws-mail
- filezilla
- kildclient
- lftp
- libpurple (pidgin)
- libsoup (NOTE: first libsoup bump then all the other gnome | gtk* apps)
 - bug-buddy
 - evolution-data-server
 - rhythmbox
 - seahorse
 - swfdec
 - vino
- liferea
- msmtmp
- net6
- python-gnutls
- weechat
- wireshark (.so)

8.6 dbus

- hal
 - evince
 - gnome-utils
 - gnome-media
 - gnome-volume-manager
 - nautilus-cd-burner
 - ivman
 - k3b
 - pmount
 - kdebase
 - xfce4-terminal
 - liferea
 - bmpx
 - bluez-libs
-

8.7 dbus-mono

- banshee
- tomboy
- f-spot
- galago-sharp

8.8 neon

- subversion
- rpm
- openoffice.org
- gst-plugins-bad
- fusedav

8.9 binutils

- amule

8.10 libtasn1

- gnutls
- evolution (need to figure out which part depends on libtasn1 ...)
- lftp
- libsoup
- loudmouth

8.11 gstreamer

(only if is an upgrade for example, from 0.8 to 0.10, or 0.10 to 0.12, etc)

- amarok
 - banshee
 - rhythmbox
 - totem
 - gnome-applets
 - gnome-control-center
 - and probably a lot of gnome too
-

8.12 gtk+2

(only need for special version bumps. Example 2.8 → 2.10 we need bump these packs because /usr/lib/gtk+-2.0/1.X.X directory changed. BTW not at all bumps. Ex.: 2.6→2.8)

- gtk+2-engines
- librsvg
- libgnomeui
- gtk-xfce-engines
- kde-theme-qtcurve

8.13 libcdio

- sound-juicer

8.14 vte

- gnome-terminal
 - xfce4-terminal
 - gtk2-sharp
 - anjuta
 - tilda
 - grip
 - awn-extras-applets
 - guake
 - mlview
 - roxterm
 - ruby-gnome2
 - gnome-desktop-sharp
 - cairo-dock-plugins
 - geany
 - sakura
 - sjterm
 - termit
 - nemiver
 - lxterminal
-

8.15 firefox

To rebuild packages for a new version, bump the up2date in source/include/firefox-118n.sh, then:

```
cd source/locale-extra/  
for i in $(ls -d firefox-*|egrep -v 'spell|dict')  
do  
    cd $i  
    bumppkg && repoman rec "- version bump"  
    cd - >/dev/null  
done
```

8.16 xulrunner

- galeon
- epiphany
- devhelp
- yelp

8.17 wireless_tools

- kdenetwork

8.18 parted

To rebuild packages for parted-1.8.8:

```
revdep-rebuild 429 --other --sed "s|'parted[^\']*'|'parted>=1.8.8'|"
```

8.19 libpqxx

- kpogre
- asterisk-addons
- asterisk
- koffice

8.20 openobex

- kdebluetooth

8.21 bluez-libs

- bluez-utils
- kdebluetooth
- libbtctl
- gnome-bluetooth
- bluez-pin

8.22 gail

(.so version bump)

- eel
- gtkhtml

8.23 imagemagick

- dvdauthor

8.24 evolution-data-server

- ekiga
- evolution

8.25 x264

- mplayer
- avidemux

8.26 ocaml

- facile

8.27 openbox

- obconf

8.28 pilot-link

- gnome-pilot
- gnome-pilot-conduits
- libmal
- kdepim
- evolution
- sypheed-claws

8.29 php

- eaccelerator
-

8.30 libevent

(on sover change)

- tor
- nfs-utils
- trickle

8.31 exiv2

- gwenview
- libkexiv2
- digikam
- kipi-plugins
- kphotoalbum

8.32 icu4c

- bmpx
- boost
- openoffice.org
- rblibtorrent
- tin
- webkit

8.33 c-ares

- aria2
- php
- bzflag
- xine-ui
- sword

8.34 libofx

- homebank

8.35 directfb

- gst-plugins-bad
 - splashy
-

8.36 sword

- bibletime

8.37 gpm

- fpc
- joe
- vim
- pycrypto
- jed
- xemacs
- fte
- links
- elinks
- aumix
- aalib

9 Creating translations for init scripts

Marcus Habermehl <bmh1980de@yahoo.de>

9.1 Preparing the source

To make a script translatable you must first add these two lines to the rc script.

```
TEXTDOMAIN=my_service
TEXTDOMAINDIR=/lib/initscripts/messages
```

To mark a string as translatable in bash you must prefix the string with \$.

```
echo $"This is a translatable string."
```

9.2 Creating the pot file

After this you must create the pot file.

```
$ bash --dump-po-strings rc.my_service | xgettext -L PO -o rc.my_service.pot -
```

9.3 Creating a po file

In the next step you create the po file.

```
$ msginit -l hu_HU
```

Now you can edit the po file with any editor.

9.4 Creating the mo files

To create and install the mo files, you must add the po files to the source() array and use the `Frcd2` macro in `build()`.

10 Frugalware AsciiDoc quickstart

Since 0.6 Frugalware, all documentation is written in AsciiDoc which means we have to write `README.Frugalware` files in AsciiDoc syntax. Here are some basic AsciiDoc features and some things you should and should not do a `README.Frugalware`.

10.1 Features

You can use `*bold*`, `_italic_` and also ``monospaced`` fonts.

You can also `“quote”` if you want to do so.

When you want to add something to the

```
-----  
# root command line  
$ user command line  
> keyboard input  
-----
```

that's no problem at all.

Maybe you want bulleted items:

```
.Items  
* item 1  
* item 2  
* here is number 3
```

And you can also create lists:

```
1. First  
+  
It's indented, belongs to first.  
+  
And this paragraph is also indented.  
  
2. Second  
+  
This is inside the second point.  
+  
2.1. Foo  
+  
2.2. Bar  
+  
a. Baz  
  
3. Third  
  
End of list.
```

Some extras:

```
NOTE: You can also place notes.
```

```
TIP: It's a tip
```

WARNING: Warning.

IMPORTANT: This is important

CAUTION: Cave canem!

10.2 Restrictions

You **must not** underline titles with = or -. You might use ~, and ^ for subchapters. If you want one line titles place 3 or 4 = before the title and a space.

10.3 Skeleton for README.Frugalwares

Your titles should look similar to this:

```
=== First chapter
-----
# pacman-g2 -Syu
-----

=== Second one

`\_F_foobar`

==== This is a subchapter...

...and its contents.
```

or

```
First chapter
~~~~~

-----

# pacman-g2 -Syu
-----

Second one
~~~~~

`\_F_foobar`

This is a subchapter...
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

...and its contents.
```

10.4 Skeleton for standalone documentation

You might ask then: okay, but how do I start? Here is a really simple example:

```
= Title
Author Name <foo@frugalware.org>

== First chapter
```

```
-----  
pacman-g2 -Syu  
-----  
  
== Second one  
  
'\_F_foobar'
```

And you can generate the HTML using

```
asciidoc -a toc -a numbered skel.txt
```

The documentation should be placed under the `/docs` dir in the FST. Please add a link to it in `index.txt` and in `index-user.txt` or `index-devel.txt` depending on the type of the documentation.

10.5 Buiding it on your own machine

Install the tools necessary to build the documentation (if you haven't already done so):

```
# pacman-g2 -S make asciidoc po4a
```

Get the necessary source code and translations:

```
$ mkdir ~/git  
$ cd ~/git  
$ git clone http://frugalware.org/git/pub/frugalware/frugalware-current current  
$ git clone http://frugalware.org/git/pub/other/translations  
$ cd current/docs
```

Generate additional documentation and update the po files from the translations repository:

```
$ make packages.txt user.txt po
```

Generate the localized documentation source from the po files:

```
$ po4a -k 0 po4a.cfg
```

Generate HTML from the source:

```
$ cd hu  
$ asciidoc -a toc -a numbered -a sectids network.txt
```

Now you can look at the result of your translation in a web browser.

If you have already done this, and you updated the translation, you need to:

```
$ cd ~/git/translations  
$ git pull --rebase  
$ cd ~/git/current/docs  
$ rm -rf po  
$ make po  
$ po4a -k 0 po4a.cfg  
$ cd hu  
$ asciidoc -a toc -a numbered -a sectids network.txt
```

and now you should be able to see your updated translation in the updated HTML.

10.6 Adding a new project to Pootle

Well, this happens rarely, and so is not well documented, but here is what is needed:

- `autogen.sh` should support importing po files from the `translations` repository and should have a `--pot-only` switch. `gnetconfig` is a good example.
- The pot file should be updated daily. Add the project's `autogen.sh` to `-current's /tools/genpkgdbs`.
- Run the above command manually once.
- Add the pot file to `pootle-update` in the `pacman-tools` repository.
- Run `pootle-update` manually once.
- Log in to Pootle with administrator rights and create a new project.
- Add the necessary new languages on the web interface.
- Translate a few strings for one language and commit.
- Pull the translations repository locally and verify that you get the expected results.

11 Frequently Asked Developer Questions

11.1 What is the recommended way to version bump a package if I don't have git push access?

- a. Update the FrugalBuild.
- b. Optional: update the patches/docs/etc.
- c. Compile the package.
- d. Upload the new `.fpm` to incoming.
- e. `repoman rec`, `git format-patch` and `git send-email` the fixes. (Don't forget to set your git identity!)

11.2 `makepkg` ends up with `<packagename>: /usr/info/dir: exists in filesystem`

Instead of

```
make DESTDIR=$startdir/pkg install
```

you should write

```
Fmakeinstall
```

in your FrugalBuild.

11.3 I can't `pacman-g2 -Su <package>`, it says local version is newer, but I know it isn't!

This is a bug in the package's version numbering, so please report this in the Bug Tracker System. Since `pacman-g2` checks the version numbers (installed vs. repo version), the new package's version must be bigger than the old one to upgrade flawlessly.

11.4 What does 5.55 SBU mean?

It took 5.55 times longer for the maintainer to compile this package than `binutils`. So if you want to know how long it will take to compile a package with 5.55 SBU, you should first compile `binutils` (`makepkg` helps you, as it writes how many seconds elapsed). Then you should multiply it by 5.55 to know how many seconds it will take to compile the package.

11.5 Why do maintainers cry about my new package's tarball?

Let's have a look at the filelist of eaccelerator's tarball:

```
$ tar -tf eaccelerator-0.9.3-1.tar.bz2
eaccelerator/
eaccelerator/eaccelerator-0.9.3.zip
eaccelerator/FrugalBuild
eaccelerator/README.Frugalware
eaccelerator/eaccelerator-0.9.3-1-i686.fpm
```

You have to name the tarball as <pkgname>-<pkgver>-<pkgrel>.tar.bz2 (or gz), which should only contain a <pkgname> directory at first level, and all the files needed to create the fpm in it. It is the easiest way for the maintainers to work with your tarball when adding your package to the repo.

11.6 What should and shouldn't I include in depends(), rodepends() and makedepends()?

You should include only what `chkdep -p` recommends, and avoid trivial makedepends, including:

- auto*
- make
- gcc
- kernel-headers
- libtool
- glibc

Don't forget: every depends is a makedepends as well!

The rodepends() array should only contain packages really needed for running the given application.

11.7 What are the various dependency-control arrays for?

- *depends* should contain any packages that this one depends on at compile and run time as well.
- *makedepends* is for packages that this one needs to compile.
- *rodepends* is for run time only dependencies; eg. a wordlist package (with no executables) needs a program which can handle it as a dictionary.
- *provides* is an alternate name for the package. Main use is for more packages which do the same; eg. hunspell-en and hunspell-de both provide hunspell-dict, and hunspell depends on hunspell-dict instead of any specific language. (Sometimes those packages are conflicting, like postfix provides *and* conflicts with mta, and exim too - this way there can be only one MTA on the system, without the need to know other MTAs' name.)

Be careful with dependency-cycles: while pacman-g2 can handle them, makepkg can not.

11.8 How can I have PHP to work with my newly packaged eaccelerator/anything extension?

Since package A should not tamper with package B's config files, you should write a README.Frugalware, describing how to enable/use the extension, include it in source() and Fdoc README.Frugalware.

11.9 How can I cross-compile (package) an architecture-independent (non-binary) program?

You should modify `carch` and `chost` in `/etc/makepkg.conf` and build the package again.

11.10 repoman upd can't create /var/fst/ as it already exists

Su - to root and

```
cd /var/fst && mv * frugalware-current
```

11.11 How can I access the central FW repo (mirrors are too slow for me)?

```
git clone http://git.frugalware.org/repos/frugalware-current
```

This creates a new local repo for you, which is a copy of the central repo. To update it, run

```
git pull --rebase
```

in it. That's all to have a read-only copy; if you want to git send-email patches, then you should read the [Git docs](#) to set up your name, email, etc.

11.12 What should I write as patch name and long comment at repoman rec?

Patch name should be the same as the fpm (but without `.fpm`, of course); and long comment should only contain what you have done to create that patch (eg. "added i686 to archs(")).

11.13 Where should I place my comments about a package?

You mean `README.Frugalware`. It should be in `source()` and then at the end of the `build()` you should use:

```
Fdoc README.Frugalware
```

It is automatically included if you use empty `build()` or `Fbuild`.

11.14 I want to work with the latest development version of pacman&co.! How?

```
$ git clone http://git.frugalware.org/repos/pacman-tools
$ cd pacman-tools
$ make dist
```

You will have a brand new `.tar.gz`. Give it to `pacman-tools`' `FrugalBuild`, correct the checksum, create a new `pacman-tools` package (`makepkg -fuch` helps) and install it. That's all (and if you don't understand this, read it again, and if it's still not clear, then wait for `pacman-tools`' normal upgrade since you don't need this really)...

11.15 Naming locale packages

What is the order of a new package's locales? How should I name them?

Have a look at `hunspell` There is a `hunspell` package, which depends on `hunspell-dict`. There is no package named `hunspell-dict`, but it is provided by the locale packages. The most important ones are `-en` (`==en_US`), `-hu` (`==hu_HU`), `-de` (`==de_DE`), `-fr` (`==fr_FR`), `-it` (`==it_IT`), `-es` (`==es_ES`) and `-sk` (`==sk_SK`). Here are others: `-en_US`, `-de_CH`, `-es_MX`.

The `-xx` packages will be installed by the non-CD based (ie. `netinst`, DVD) installers.

11.16 Error handling

You are responsible for checking if a command used in `build()` fails. The best is to use the `F*` macros where possible since they handle the errors for you. If you need custom commands, it's recommended to append `|| return 1` to every line, so that `build()` will stop if an error occurs.

11.17 Permissions

If text files (header files, documentation) are executable, feel free to fix their permission. A bigger problem is the permission of the shared libraries. They must be executable, please fix their permission if necessary. As always, it's recommended to create a patch to fix the problem and send it to the upstream project.

11.18 Stripping

Stripping binaries is unnecessary and pointless. Unless you use `options=(\nostrip\)` in the `FrugalBuild`, it's done by `makepkg` automatically.

11.19 When should I use `$Fsrcdir` and `$Fdestdir`

Most `F*` macros will prepend/append those variables for you, but if you use custom commands, then you always have to use them.

11.20 When should I increment a package's release number?

- If your change affects only the `FrugalBuild` (like an `up2date` fix) then you should not, just push your change.
- If your change affects the `fdb` or the `fpm` (change in `build()`, `depends()` fix, etc) you should do so.

11.21 How do I repair a corrupted package database?

Restore a backup from the `/pub/other/fdb-snapshot` directory, and check its version (the `.version` file in the tarball).

Then run:

```
$ for i in `git log --pretty=oneline 94a41e0..|sed 's/^[^ ]* \([^ ]*\).*\/\1/'\`  
|sed 's/-[^-]*-[^-]*-[^-]*$//'\`; do ls ../source/*/${i} &>/dev/null \  
|| continue; updatesync upd frugalware-current.fdb \  
../source/*/${i}/FrugalBuild; done
```

12 Frugalware Source Tree Testsuite

12.1 Introduction

The testsuite is a set of several simple unit tests. Most of the tests were written when a typo was found, so that we hope next time it'll be detected automatically. When a problem was found, a test was created and the test failed. After the problem was fixed the test passed. The statistics section contains special tests: we are aware that they do not pass, but their actual output is interesting for us. The output of the testsuite is sent to the `frugalware-devel@` mailing list daily.

Since the tests in the testsuite section should pass, if one fails it is expected to be fixed within a day, especially if your name is listed next to a line.

You can find the tests under the `/t` directory of `FST`, the statistics are under `/t/s`.

12.2 Rules

Basically there are 3 simple rules for these tests:

- If the first argument is `--help`, they should print a short (less than 80 chars) description. This will be displayed if the test fails as sometimes the name of the test may not be descriptive enough.
- The tests are called in a `./testname` form, without any argument. This allows you to use various interpreted programming languages (python, bash, etc.).
- If the test *passes*, there should be no output. This means that there may be a `-v` or `--verbose` option to generate output even if the test passes, that's not a problem. If the test *fails* there must be some output. For example if there are problematic packages, then it's recommended to list each package in a separate line with their path under FST.

12.3 Technical details

Given that all the files in the fdb and fpm files are owned by root, if you want to operate on them, then you need to use fakeroot. The testsuite wrapper won't do this for you. A common practice is to write a generic python script that operates on the fdb, then create a shell wrapper for each arch, which will call the python script via fakeroot.

13 Translations

13.1 Introduction

Localization is important for every user who doesn't speak English fluently. If your native language is not English, then you can help us by translating a few sentences to your native language. If you would like to help, the following steps are necessary:

- Visit the [web interface](#) and register.
- Select your language (ie. if you would like to contribute French translation, select French). If your language is not listed, then ask for addition on our developer mailing list.
- Select what projects you would like to translate. It's good to start with some smaller project like the homepage or the setup. If the given project has no `.po` file for your language, contact us.
- Now you can begin translating, but your changes won't hit the master repo, you need additional permissions to commit from the sandbox. Ask us for commit access.

A few tips if you're new to pootle:

- By default you can edit the whole translation, but usually you would like to see only the untranslated and fuzzy strings. You can search for them by clicking on "Show editing functions" then selecting "Quick translate".
- You can commit a po file by clicking on "Show editing functions" then selecting "Commit".
- You can search for fuzzy translations by clicking on "Show editing functions", selecting "Show checks" and then the "isfuzzy" check.

13.2 Rules

There are not many, at the moment.

- Please don't translate the `== NAME` and `== SYNOPSIS` strings in the manpages, docbook does it already and asciidoc fails to create the manpage if it's already translated.
 - The first translator for a language (this can be changed if requested) receives all rights for a given project, except: Suggest, Overwrite, Assign, Administrate.
-

13.3 Goals

When we created the current mechanism of handling translations, we had the following goals:

- When we modify source code or documentation, the translators should be able to begin the necessary (if any) translations without any manual action.
- It would be nice to overview the localization status of a language.
- It should be easy to maintain the translation (ie. doing a manual sync for big documents is rather problematic).
- Translators are not developers, write access to the translations should not require any other access right.
- It should be possible for anyone to translate, but only given users should be able to push changes.

13.4 Overview

Now let's see how all this is possible. We'll take our `asciidoc` documentation as an example.

First, we need to extract the translatable strings from the sources. This is an important step since this way a document is split into paragraphs and you can then later translate even a single paragraph rather than choosing between translating a 10-page-length document entirely or not. We use `po4a` for this purpose. It creates a template, named `docs.pot`, which is transferred daily to the translation server.

(For source codes we usually use the `intltool-update` utility to extract translatable strings.)

Right after the transfer, the `po` localization files are updated using `msgmerge` from the `gettext` package: this way the translators do not have to re-translate the strings which are already done.

On that machine, we use a web interface for the translation. This has several advantages:

- The translators can register and begin their work without any confirmation from our developer team.
- Those accounts are - of course - not real unix accounts but just virtual ones.
- We can give commit access for users by specifying their project and language. So everybody can make translations but only users we know can push the changes.
- Collaboration for people who do not know what a patch or a version control system is now should not be a big problem. This is important since for example the whole documentation is one big file per language.

Once a user with enough privileges pushes the translation to our `git` version control system, we can use it. The documentation is built daily and we pull the new translations from the dedicated repo before each build.

The output of the English build is available [here](#). If it contains any error or warning, the testsuite will let us know by including them in the daily testsuite mail, sent to the developer mailing list. The log of the localization builds is available [here](#).

There we use `po4a` again to reconstruct the original (now in some language other than English) document from the translated strings.

For source code we pull the translations right before creating a release tarball so. This has the following benefits:

- We ship the latest translations
- Once the tarball is ready, users who would like to compile the source code should not fetch the translations manually.

The proof of concept for this mechanism is our French documentation which is more than 80 pages length and includes zero percent of manual editing by the developers (while till now we had to push the submitted - by email and other undocumented channels - translated documents manually, hoping that the newer version is better than the old was).

14 How to port Frugalware to a new architecture

14.1 Introduction

This document is a draft about how to port Frugalware to a new architecture.

14.2 Toolchain

- Install any existing distro to the given architecture. No matter what kind of it, but make sure you install the normal development tools like header files, gcc, make, etc.
- Compile from source (based on the FrugalBuilds) our development tools like pacman-g2, pacman-tools (+ deps: libarchive, etc if they are not available.)
- Build a minimal toolchain: binutils, gcc, glibc (in this order) outside chroot, with dep checking disabled (makepkg -dHcu).
- Build packages which are necessary to build in chroot: see the COREPKGS variable in /etc/makepkg.conf (same makepkg switches).

Given that repoman won't allow you to upload which are not built in chroot, here is a simple script to upload and register then till you don't have a chroot:

```
#!/bin/sh
scp *.fpm genesis:git/current/frugalware-ppc
pkgname=$(pwd|sed 's|.*||')
ssh genesis "cd git/current/frugalware-ppc; arch=ppc updatesync upd frugalware-current.fdb <-
  ../source/*/$(pkgname)/FrugalBuild"
```

Replace genesis with the server name and git/current with an other path if you don't have such a symlink in your HOME.

Now you can start building in chroot and uploading real packages.

Note

Yes, this means that you have to build the toolchain twice. Also known as *bootstrapping*.

14.3 Base system

You should start porting with packages from the *base* category, once you are done with it, you should be able to install (manually) a bootable system, after manually configuring a boot manager.

14.4 The rest

That depends on your needs, you can port additional packages as well.

15 GNU Free Documentation License

Version 1.2, November 2002

```
Copyright (C) 2000,2001,2002 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

15.1 PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

15.2 APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned

below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

15.3 VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

15.4 COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

15.5 MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

15.6 COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or

else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

15.7 COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

15.8 AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

15.9 TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

15.10 TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

15.11 FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.
