Fedora Draft Documentation

Documentation Guide

A how-to guide for creating and maintaining documentation for Fedora

Eric Christensen

Jared Smith

Joshua Wulf
A how-to guide for creating and maintaining documentation for Fedora Edition 0.2

Author Eric Christensen sparks@fedoraproject.org
Author Jared Smith jsmith@fedoraproject.org
Author Joshua Wulf jwulf@redhat.com

Copyright © 2011 Fedora Project Contributors.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at http://creativecommons.org/licenses/by-sa/3.0/. The original authors of this document, and Red Hat, designate the Fedora Project as the "Attribution Party" for purposes of CC-BY-SA. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, MetaMatrix, Fedora, the Infinity Logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

For guidelines on the permitted uses of the Fedora trademarks, refer to https://fedoraproject.org/wiki/Legal:Trademark_guidelines.

Linux® is the registered trademark of Linus Torvalds in the United States and other countries.

Java® is a registered trademark of Oracle and/or its affiliates.

XFS® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

All other trademarks are the property of their respective owners.

A how-to guide for creating and maintaining documentation for Fedora.
# Preface

1. Document Conventions ................................................................. v
   1.1. Typographic Conventions ...................................................... v
   1.2. Pull-quote Conventions ....................................................... vi
   1.3. Notes and Warnings .............................................................. vii

2. We Need Feedback! ...................................................................... vii

# 1. Introduction

1.1. Purpose of This Guide ............................................................ 1
1.2. Tips for Success ...................................................................... 1
   1.2.1. Be Bold! ........................................................................ 1
   1.2.2. Don't Let the Tools Scare You ........................................ 1
   1.2.3. There's a Team to Back You Up ...................................... 1
1.3. Organization of This Guide .................................................... 1

# 2. Documentation Workflow

2.1. Write First, Format Later ....................................................... 3
2.2. Formatting in DocBook .......................................................... 3
2.3. Pushing Strings to Translation .............................................. 3
2.4. Pulling Strings from Translation .......................................... 3
2.5. Building Documents ............................................................. 3
2.6. Publishing Documents to the Web ....................................... 3
2.7. Building Packages from Documents .................................... 3
2.8. Creating a patch ................................................................. 3

# 3. Brief Introduction to DocBook

   3.1.1. DocBook as an XML Markup Language ......................... 9
   3.1.2. Human-readable Tags .................................................... 9
   3.1.3. Text-based Format for Easy Revision Control ................. 9
   3.1.4. Separation Between Content and Style ......................... 9
   3.1.5. Output in a Variety of Formats ..................................... 9
3.2. Parts of a DocBook File ......................................................... 9
   3.2.1. Entities ........................................................................ 9
3.3. Dividing a Document into Multiple Files with XIncludes .... 10
3.4. Entities: With Great Power Comes Great Responsibility .... 10

# 4. XML Tools

4.1. XML Concepts ....................................................................... 11
4.2. xmlfmt ................................................................................ 11
4.3. xsltproc ............................................................................. 11
4.4. xmldump ............................................................................. 11

# 5. Publican

5.1. Installing Publican ............................................................... 13
5.2. Creating a New Document .................................................. 13
5.3. Validating XML Code ......................................................... 14
5.4. Building a Document .......................................................... 14
5.5. Publishing a Document ......................................................... 15
   5.5.1. Preparing Your System ................................................ 15
   5.5.2. Publishing a Document on the Web ......................... 16
   5.5.3. Removing a Document from the Web ....................... 17
   5.5.4. Updating a Document on the Web ............................... 17
   5.5.5. Marking a Documentation as Draft ......................... 17
5.6. Building RPM Packages ...................................................... 18

# 6. Translations

19
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1. Translating Documentation</td>
<td>19</td>
</tr>
<tr>
<td>6.2. Introducing Transifex.net</td>
<td>19</td>
</tr>
<tr>
<td>6.3. Creating an Account on Transifex.net</td>
<td>20</td>
</tr>
<tr>
<td>6.4. Installing the Transifex Client</td>
<td>20</td>
</tr>
<tr>
<td>6.5. Configuring .transifexrc</td>
<td>20</td>
</tr>
<tr>
<td>6.6. Initializing Transifex in Your Git Repository</td>
<td>20</td>
</tr>
<tr>
<td>6.7. Mapping Language Codes</td>
<td>20</td>
</tr>
<tr>
<td>6.8.Editing File Filters</td>
<td>21</td>
</tr>
<tr>
<td>6.9. Updating POT Files</td>
<td>22</td>
</tr>
<tr>
<td>6.10. Pushing and Pulling Translations</td>
<td>22</td>
</tr>
<tr>
<td>6.11. Branching and Updating Translations Before a New Release</td>
<td>22</td>
</tr>
<tr>
<td>6.12. Additional Resources</td>
<td>23</td>
</tr>
<tr>
<td>7. Style Guide</td>
<td>25</td>
</tr>
<tr>
<td>7.1. General tips</td>
<td>25</td>
</tr>
<tr>
<td>7.2. Abbreviations, acronyms, and initialisms</td>
<td>25</td>
</tr>
<tr>
<td>7.3. Pictures and screen captures</td>
<td>25</td>
</tr>
<tr>
<td>7.4. Software</td>
<td>25</td>
</tr>
<tr>
<td>A. Revision History</td>
<td>27</td>
</tr>
<tr>
<td>Index</td>
<td>29</td>
</tr>
</tbody>
</table>
Preface

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the Liberation Fonts\(^1\) set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

**Mono-spaced Bold**

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keycaps and key combinations. For example:

To see the contents of the file `my_next_bestselling_novel` in your current working directory, enter the `cat my_next_bestselling_novel` command at the shell prompt and press Enter to execute the command.

The above includes a file name, a shell command and a keycap, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from keycaps by the hyphen connecting each part of a key combination. For example:

Press Enter to execute the command.

Press `Ctrl+Alt+F2` to switch to the first virtual terminal. Press `Ctrl+Alt+F1` to return to your X-Windows session.

The first paragraph highlights the particular keycap to press. The second highlights two key combinations (each a set of three keycaps with each set pressed simultaneously).

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in *mono-spaced bold*. For example:

File-related classes include `filesystem` for file systems, `file` for files, and `dir` for directories. Each class has its own associated set of permissions.

**Proportional Bold**

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

Choose System → Preferences → Mouse from the main menu bar to launch Mouse Preferences. In the Buttons tab, click the Left-handed mouse check box and click

---

\(^1\) [https://fedorahosted.org/liberation-fonts/](https://fedorahosted.org/liberation-fonts/)
Close to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).

To insert a special character into a gedit file, choose Applications → Accessories → Character Map from the main menu bar. Next, choose Search → Find… from the Character Map menu bar, type the name of the character in the Search field and click Next. The character you sought will be highlighted in the Character Table. Double-click this highlighted character to place it in the Text to copy field and then click the Copy button. Now switch back to your document and choose Edit → Paste from the gedit menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

Mono-spaced Bold Italic or Proportional Bold Italic

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

To connect to a remote machine using ssh, type ssh username@domain.name at a shell prompt. If the remote machine is example.com and your username on that machine is john, type ssh john@example.com.

The mount -o remount file-system command remounts the named file system. For example, to remount the /home file system, the command is mount -o remount /home.

To see the version of a currently installed package, use the rpm -q package command. It will return a result as follows: package-version-release.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

Publican is a DocBook publishing system.

1.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in mono-spaced roman and presented thus:

<table>
<thead>
<tr>
<th>books</th>
<th>Desktop</th>
<th>documentation</th>
<th>drafts</th>
<th>mss</th>
<th>photos</th>
<th>stuff</th>
<th>svn</th>
</tr>
</thead>
<tbody>
<tr>
<td>books_tests</td>
<td>Desktop1</td>
<td>downloads</td>
<td>images</td>
<td>notes</td>
<td>scripts</td>
<td>svgs</td>
<td></td>
</tr>
</tbody>
</table>

Source-code listings are also set in mono-spaced roman but add syntax highlighting as follows:

```java
package org.jboss.book.jca.ex1;
import javax.naming.InitialContext;
```
public class ExClient{
  public static void main(String args[])
    throws Exception{
    InitialContext iniCtx = new InitialContext();
    Object ref = iniCtx.lookup("EchoBean");
    EchoHome home = (EchoHome) ref;
    Echo echo = home.create();

    System.out.println("Created Echo");
    System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
  }
}

1.3. Notes and Warnings
Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

**Note**

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

**Important**

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled 'Important' will not cause data loss but may cause irritation and frustration.

**Warning**

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

2. We Need Feedback!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla: http://bugzilla.redhat.com/bugzilla/ against the product Documentation.

When submitting a bug report, be sure to mention the manual's identifier: documentation-guide

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.
Introduction
This chapter will introduce you to some of the best practices that the Fedora Documentation Project uses to help its writers be most productive.

1.1. Purpose of This Guide

1.2. Tips for Success
Before we dive into the mechanics of the documentation, let's look at some important points that will help ensure your success in your documentation efforts.

1.2.1. Be Bold!
Although it might seem a bit trite at first glance, the most important part of writing documentation is writing. There's no need to ask permission before writing... just start writing!

1.2.2. Don't Let the Tools Scare You
The Fedora documentation group uses a number of different tools, but please don't be intimidated by the tools. A bit of practice and mentoring may be necessary to get proficient with the tools. In the meantime, just write in a text editor of your choice, and the text can always be converted into the proper format at a later time.

We will discuss the various tools in Chapter 4, XML Tools.

1.2.3. There's a Team to Back You Up
Don't be afraid to ask for assistance. The Fedora documentation team is more than willing to help you both with writing and with learning the tools and workflow. If you're on IRC, join the #fedora-docs channel on the Freenode network. We also have a mailing list where we discuss various issues related to documentation. For more information on the joining the mailing list, visit https://lists.fedoraproject.org/mailman/listinfo/docs.

1.3. Organization of This Guide
Documentation Workflow

This chapter will introduce you to the basic workflow of documents as they’re written, converted, translated, and published.

2.1. Write First, Format Later

2.2. Formatting in DocBook

2.3. Pushing Strings to Translation

Pushing strings to translation is discussed in Chapter 6, Translations.

2.4. Pulling Strings from Translation

Pulling strings from translation is discussed in Chapter 6, Translations.

2.5. Building Documents

Building documents is discussed in Section 5.4, “Building a Document”.

2.6. Publishing Documents to the Web

Publishing documents to the web is discussed in Section 5.5, “Publishing a Document”.

2.7. Building Packages from Documents

Building packages from documents is discussed in Section 5.6, “Building RPM Packages”.

2.8. Creating a patch

Patches are files that show changes between two versions of a file. The patch file can be shared via email or in a bug report, easily reviewed, and applied without requiring additional manual editing. This section details the procedure for creating a patch for a Fedora Guide.

Example 2.1. An example of a patch file

A patch file might look like this one.

```
From a6419e00b515765085cc4e646c3822ebdda607df Mon Sep 17 00:00:00 2001
From: Pete Travis <immanetize@fedoraproject.org>
Date: Mon, 30 Dec 2013 21:52:57 -0700
Subject: [PATCH] Update commands for starting, stopping, and checking the status of abrt to use systemctl instead of service/chkconfig

---
en-US/Automatic_Bug_Reporting_Tool_ABRT.xml | 20 ++++++++++----------
1 file changed, 10 insertions(+), 10 deletions(-)

index 1093dc8..f81bb33 100644
--- a/en-US/Automatic_Bug_Reporting_Tool_ABRT.xml
```
Let us examine how this patch file was created. In the following procedure, we will create a patch to correct a line in the *System Administrators Guide* to update from the legacy `service` to the current `systemctl` command.
Procedure 2.1. Patch creation workflow

1. First, find the guide you want to work on in the table at https://fedoraproject.org/wiki/Docs_Project_guides_table.

2. The repo column of the table has a link to the guide’s git repository. Open the link, and you’ll see the Clone section at the bottom of the page, listing the repository’s URLs.

3. If you have set up ssh keys for your FAS account, right click the URL that begins with ssh:// and select Copy Link Location.

   If you have not set up ssh keys in FAS, or simply want to clone anonymously, right click the url that begins with https:// and select Copy Link Location.

   It is a good idea to use the ssh:// url, even if you do not currently have commit access. If you later gain commit access, you won’t have to go back and change the upstream location of the repo.

4. In a terminal window, change to the directory you have chosen to store guides in and clone the repo. You can paste into the terminal window by right clicking it and selecting Paste. Some terminal applications will also paste with the key combination Shift+Insert.

   ```
   cd ~/fedora-docs
   git clone ssh://git.fedorahosted.org/git/docs/system-administrators-guide.git
   cd system-administrators-guide
   ```

5. If you have already cloned the repo, it is important to make sure your local copy is updated and matches the remote copy.

   Pull in new changes from remote:

   ```
   git pull
   ```

   If you have made changes in the master branch that aren’t committed, you can stash them. The files will returned to the unchanged state, and you can come back to them later. For detailed information, refer to man git-stash.

   ```
   git stash
   # continue with patch process, then bring back your stashed state with:
   git stash pop
   ```

   If you have made and committed changes in the master branch, you should push your changes:

   ```
   git push
   ```

   If you don’t have access to push commits, or you don’t want to push them, you can reset your local copy to match remote:

---

1 https://admin.fedoraproject.org/accounts
6. A new branch of the repository will allow you to make changes to the side, without affecting the master branch until we are ready to. For smaller patches, this branch is temporary and you won't have to share it. Name the branch something recognizable to help you keep track of its purpose, then check out the branch.

```
git branch abrt-systemctl-not-service
git checkout abrt-systemctl-not-service
```

7. Find the files you need to change. A bug report might say "The abrt section should use the systemctl command instead of service", but it will not tell you what file or section to edit. In DocBook, commands are usually wrapped in the `<command>` tag, so we can use that to help our search.

```
grep "<command>service" en-US/*.xml
```

```
en-US/Automatic_Bug_Reporting_Tool_ABRT.xml:
    <screen>~
    <command>service
    abrt status</command>
```

```
en-US/Automatic_Bug_Reporting_Tool_ABRT.xml:
    If the <command>service</command> command returns the <computeroutput>abrt is stopped</computeroutput> message, the daemon is not running. It can be started for the current session by entering this command:
    <screen>~
    <command>service abrt start</command>
```

8. The `grep` command has found the file we want to edit, `en-US/Automatic_Bug_Reporting_Tool_ABRT.xml`. Open the file in your favorite editor and make the required changes:

```
vim en-US/Automatic_Bug_Reporting_Tool_ABRT.xml
```

or:

```
emacs en-US/Automatic_Bug_Reporting_Tool_ABRT.xml
```

or:

```
gedit en-US/Automatic_Bug_Reporting_Tool_ABRT.xml
```

The editor you choose is not important, but the file must be saved in plain text format.

Change only the content you need to, so that the diff will be more readable. Resist the temptation to rephrase passages unless they are misleading, to avoid making more work for translators. Do not adjust sections to your preferred indentation depth; this changes no actual content and makes the patch more difficult to interpret.

9. Test your changes by building the document, and reviewing the section in the browser.

```
publican build --langs en-US --formats html
```
10. After editing, if you want to check what files you have changed, use this command:

```
$ git status
# On branch abrt-systemctl-not-service # Changes not staged for commit:
# (use "git add <file>..." to update what will be committed)
# (use "git checkout -- <file>..." to discard changes in working directory)
# # modified: en-US/Automatic_BugReporting_Tool_ABRT.xml
# no changes added to commit (use "git add" and/or
# "git commit -a")
```

11. Commit your changes. If you only made a few changes, place them all in the same commit. If you make many changes, edit different files, or work on varied issues, spread your changes out into individual commits. This makes it easier for others to follow your work.

```
$ git commit en-US/Automatic_BugReporting_Tool_ABRT.xml
```

After you run the command above, an editor will open for you to enter a commit message. Be descriptive, explain what sections you have updated and why, and provide references if available. If you are working on a specific bug report, you should also cite the bug in the commit message. Something like `RHBZ#439858` is enough but you can also cite the full URL if you prefer.

### Default Editor

The default editor used by `git` is `vi`. You can change this setting by adding a line in `~/.bashrc`, such as:

```
export EDITOR="/usr/bin/emacs"
```

You can also bypass the editor and enter your message directly with the commit command, like this:

```
$ git commit en-US/Automatic_BugReporting_Tool_ABRT.xml -m "Update commands for starting, stopping, and checking the status of abrtd to use systemctl instead of service/chkconfig"
```

12. Check the status of your repository again, and make sure all the intended changes have been committed.

```
$ git status
# On branch abrt-systemctl-not-service nothing to commit, working directory clean
```

13. Now create your patches. Each commit will create one patch file, showing the differences between your changes and the master branch.
Now that you've created the patch, share it! Attach it to the bug ticket, send it to the mailing list, or share it with your mentor. The patch needs to get out for review and inclusion in the master branch.

If there isn't an existing bug, you can create a new one. The patch created when writing this section is attached to https://bugzilla.redhat.com/show_bug.cgi?id=1047428

After you have shared your patch, you can clean up. Checking out the master branch will return the files to the unedited state:

```
git checkout master
```

Clean any files that git is not tracking from the directory. This will delete your patches, so make sure you have sent them off for review.

```
git clean -xdf
```

You can delete the local working branch as well. Again, make sure you have shared your patches, or your work will be lost.

```
git branch -D abrt-systemctl-not-service
Deleted branch abrt-systemctl-not-service (was a6419e0).
```

After review, the patch can be downloaded and then applied using `git`:

```
git apply 0001-Update-commands-for-starting-stopping-and-checking-t.patch
```
Brief Introduction to DocBook

This chapter will give you a brief introduction to DocBook.

3.1. Why DocBook?
DocBook has many features which make it suitable for documentation in Fedora and other projects.

3.1.1. DocBook as an XML Markup Language

3.1.2. Human-readable Tags
DocBook tags are generally self-explaining. They are named after what they define, even though shorter terms might make more efficient use of space. This makes it easy for writers and editors to quickly look at a document and understand how it is constructed.

3.1.3. Text-based Format for Easy Revision Control
Version control provides two critical features for collaborative documentation work: concurrent editing and history. The text-based format of DocBook makes it well-suited to being managed by a version control system.

3.1.4. Separation Between Content and Style
The DocBook standard separates a document's content and its formatting. The writer uses XML tags to structure the elements of document, and the formatting can be changed independently later. A change to the formatting of an element does not require changing every instance of that element.

3.1.5. Output in a Variety of Formats
DocBook is a source format. The publican tool is used to generate rendered documents in several different formats. publican can generate HTML, PDF, and epub formats from the same source file.

3.2. Parts of a DocBook File

3.2.1. Entities
Entities are like variables. They allow for one change to be reflected many times in a document. Entities are usually defined early in the document using the following syntax:

```
<!ENTITY ENTITYNAME "value">
```

Entities are then referred to in the document source with &ENTITYNAME;. The most common use of entities are for phrases like the current version of a product or the product name.
3.3. Dividing a Document into Multiple Files with XIncludes

As a document grows, managing it as a single file becomes challenging. In addition, having multiple files compose a single document allows for some files to be reused between documents, reducing the overhead for a large documentation package. Fortunately, DocBook supports including multiple files. To include an additional file into a document, refer to the following example:

```xml
&lt;xinclude xmlns:xsi="http://www.w3.org/2001/XInclude" xsi:href="Included_File.xml" />
```

When the document is rendered, the included text will appear as if it were inserted entirely in the main file.

3.4. Entities: With Great Power Comes Great Responsibility
XML Tools
This chapter will walk you through a few of the commonly-used XML tools that will make it easier to work with DocBook files.

4.1. XML Concepts

4.2. xmllint

4.3. xsltproc

4.4. xsltwords
Publican

Publican is a program that creates the end product that our customers see. The program creates html, html-single, PDF, and ePUB formatted files from DocBook XML source code. The files that Publican builds can then be uploaded to a website or packaged in an RPM.

Publican is a command line interface (CLI) program that uses several commands for creating user files from DocBook XML source code. Publican will take care of several tasks for you including building a new book or article, build the book in several formats, and create an RPM package that can be used to install the document locally onto a computer.

Publican uses modules called brands to add logos and color schemes to documents produced for a specific project, and thus ensures that the look of these documents is consistent. At the top right of every HTML page, the Fedora brand includes a link back to the Fedora documentation site, from a graphic containing the English word “Documentation”. We can localise this graphic so that the word “Documentation” appears in the same language into which the document itself is translated.

5.1. Installing Publican

To install Publican itself and the fedora brand that is used by the Fedora Documentation Project, type the following at a shell prompt as root:

```
yum install publican publican-fedora
```

By default, only the common brand is installed with the publican package. There are many brands in Fedora repositories, such as oVirt (publican-ovirt), JBoss (publican-jboss), or genome (publican-genome). For a full list of available brands, refer to the Publican Users’ Guide.

5.2. Creating a New Document

When creating a new document, Publican allows you to choose between a book and an article. Compared to books, articles have a simpler structure, do not have a preface or a separate title page, and generally tend to be much shorter. Use an article if you intend to write a text with only few pages. For a larger documents, it is usually better to use a book.

To create a new book, type the following at a shell prompt:

```
publican create --type book --name "Document Name" --brand brand
```

where *Document Name* is the document title and *brand* is the brand you want to use (this is typically *fedora*). Similarly, run the following command to create an article:

```
publican create --type article --name "Document Name" --brand brand
```

Additionally, you can specify the --lang option followed by a language code (such as pt-BR or cs-CZ) to write the book in a different language than English.

The publican create command creates a new directory named *Document_Name*. This directory contains the publican.cfg configuration file and another directory named after the language code (that is, en-US/ by default), which contains boilerplate files as described in Table 5.1, “Files created by Publican”.

**5.3. Validating XML Code**

*Hic sunt leones.*

**5.4. Building a Document**

To build a document in order to review it, use the following command:

```
publican build --langs languages --formats formats
```

where `languages` is a comma-separated list of language codes and `formats` is a comma-separated list of file formats you want to build as described in Table 5.2, "Available target formats".

By default, the `publican build` command stores created files in the `tmp/` directory where you can review the output. To build a final version of the document, add the `--publish` option as follows:

```
publican build --publish --langs languages --formats formats
```

This places the generated files in the `publish/` directory. If you intend to publish the document at the [http://docs.fedoraproject.org](http://docs.fedoraproject.org) website, use also the `--embedtoc` option. For detailed information on how to publish a document on the Fedora Documentation Project website, refer to Section 5.5, "Publishing a Document".

**Table 5.2. Available target formats**

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>html</code></td>
<td>Generates multiple HTML pages.</td>
</tr>
<tr>
<td><code>html-single</code></td>
<td>Generates a single HTML page.</td>
</tr>
<tr>
<td><code>html-desktop</code></td>
<td>Generates a single HTML page with a table of contents in a sidebar.</td>
</tr>
<tr>
<td><code>pdf</code></td>
<td>Generates a PDF file.</td>
</tr>
<tr>
<td><code>epub</code></td>
<td>Generates an EPUB file.</td>
</tr>
<tr>
<td><code>txt</code></td>
<td>Generates a plain text file.</td>
</tr>
<tr>
<td>Format</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>man</td>
<td>Generates a manual page.</td>
</tr>
<tr>
<td>eclipse</td>
<td>Generates an Eclipse help plug-in.</td>
</tr>
<tr>
<td>test</td>
<td>Does not generate any files, only validates the XML files.</td>
</tr>
</tbody>
</table>

5.5. Publishing a Document

Publishing a document to the Fedora Documentation website isn't that difficult when Publican is doing the heavy lifting. It is assumed that you already have the repository cloned on the computer that Publican will be working from and that your configuration is complete.

![Warning]

Documents produced with versions of Publican prior to 2.5 are not compatible with the current website structure. Always make sure that you have the latest version of Publican installed before you publish documents.

5.5.1. Preparing Your System

Before we start building your documents into the proper formats for user consumption we need to pull down the webpage source which you'll later add your document into and commit. The following procedure will instruct you on how to do this.

- Create a local copy of the git repository of the docs.fedoraproject.org website: `git clone ssh://USERNAME@git.fedorahosted.org/git/docs/web.git` where `USERNAME` is your FAS username.

  ![Note]

  This download will take some time, even on fast connections.

- Change into the directory into which you downloaded the `web.git` repo, and make a copy of `homepage.tmp` named `homepage.cfg.cp` `homepage.tmp` `homepage.cfg`

- Edit the `homepage.cfg` file to provide the absolute paths to the `fedoradocs.db` file and the `public_html` directory. For example, if you downloaded the `web.git` repo to the `fedoradocs` subdirectory of your home directory, your homepage.cfg file might look like:

  ```
db_file: /home/jsmith/fedoradocs/web/fedoradocs.db
toc_path: /home/jsmith/fedoradocs/web/public_html
host: http://docs.fedoraproject.org
title: "Fedora Documentation"
search: 
```

- Make sure you have the latest version of Publican and the Fedora brand package. As root, run: `yum update publican publican-fedora`
5.5.2. Publishing a Document on the Web

**Warning**

Publican now controls the directory structure and the SQLite database that manages the site and its tables of contents. Do not add or remove directories from the directory tree manually as we did in the past.

**Warning**

If you are publishing draft documentation, be sure to follow the instructions described at Publishing draft documentation first.

**Warning**

If you are publishing the final version of documentation that was first published in the "Draft Documentation" category, be sure to follow the instructions described at Unpublishing draft documentation first.

**Incorrect product or version**

If the publish results in an incorrect product or version, check the `Type_Info.xml` to ensure the `productname` and `productversion` elements match the desired output. If they are correct, check the `publican.cfg` file for lines that begin with `version` or `web_version_label`. Delete those lines, if they exist.

- Update your copy of the `docs.fedoraproject.org` website. In the directory where you keep your local copy of the site, run: `git pull`

- Change to the directory where you keep a checked-out copy of the document that you want to publish, then run: `publican build --embedtoc --publish --formats epub,html,html-single,pdf --langs LANGUAGE_CODES` where `LANGUAGE_CODES` is a comma-separated list of the languages in which you want to publish this document.

- Browse to the publish subdirectory and to the documents themselves inside it to ensure that the documents have built as you expected. In particular, verify the product name is `Fedora` (note capitalization), the version number is correct, the document title is properly capitalized and spaced: for example, `Foo Guide`, not `foo-guide`

- Install the book to the Fedora website: `publican install_book --site_config PATH_TO_SITE_CONFIG_FILE --lang LANGUAGE_CODE` where
5.5.3. Removing a Document from the Web

To remove a document for a particular version of Fedora in a particular language:

- Update your copy of the docs.fedoraproject.org website. In the directory where you keep your local copy of the site, run: `git pull`
- Change to the directory where you keep a checked-out copy of the document that you want to remove, then run: `publican remove_book --site_config PATH_TO_SITE_CONFIG_FILE --lang LANGUAGE_CODE` where `PATH_TO_SITE_CONFIG_FILE` is the path to the `homepage.cfg` file in your local copy of the docs.fedoraproject.org website, and `LANGUAGE_CODE` is the language in which you are removing the document. Note that you can only run `publican remove_book` for one language at a time.

- Change to the directory in which you keep your local copy of the site and run:

  ```
  git add .
  git commit -m "DESCRIPTION_OF_YOUR_CHANGES"
  git push
  ```

- Publican can only remove the local files and directories on your system, not their record in Git. To remove the deleted files from your local Git repo, run: `for f in $(git ls-files --deleted); do git rm $f; done`. Then push these changes to the remote repo: `git commit -m "rm unused files" git push`

**Warning**

A `git rm` command gone wrong can cause widespread damage to the documentation site. If you are not absolutely sure of what you are doing, ask for help.

5.5.4. Updating a Document on the Web

To update a document previously published in a particular language for a particular version of Fedora, change into the directory in which you keep a checked-out copy of the document, then run `publican install_book`, as if you were installing it for the first time. Publican automatically replaces the old version with the new one.

5.5.5. Marking a Documentation as Draft

Release early, release often.
Documents can be released before they are completely ready for the users. When this happens it's important to mark and publish the document as a draft. To do this there are three things that need to be done so that everyone knows that the guide isn't complete and ready for public consumption.

The first is setting the draft switch in the document itself. To do this you need to add `status="draft"` to your `Doc_Name.xml` file in the `book` or `article` heading so it looks like `book status="draft"`. This provides a draft watermark on all pages of the text.

The next task is to modify your `Book_Info.xml` or `Article_Info.xml` file. Change the `productname` tag to Fedora Draft Documentation and remove the contents of `productnumber`. Optionally, if you plan to release the document for a particular Fedora release, include the Fedora release number in the `edition` tag. For example, if you plan to release the book for Fedora 14, you would set edition to: `14.0.1`. Here 14.0.1 signifies Fedora 14, edition 0.1.

Lastly, in the `Publican.cfg` add the following lines: `version: 0.1` and `web_version_label: UNUSED`.

After completing the above tasks your guide will be sufficiently marked as draft and can safely be published to the Docs website.

5.6. Building RPM Packages

*Hic sunt leones*
Translations

Translations are an important piece of the documentation process. If users cannot read the native language that the documentation is written in, then the documentation is not helpful to those users and the hard work that went into the documentation is not being utilized to its fullest extent.

This chapter discusses how translations are provided in Fedora documentation.

6.1. Translating Documentation

The cornerstone of translation when using XML files such as DocBook is the Portable Object or PO file. PO files provide a way for independent individuals or teams to translate documents.

PO files are usually found in two forms:

- the .pot file, or PO Template (POT)
- the .po file, or translation file (PO)

A POT is, as the name suggests, a blank template for new translations. It contains some header information and a list of strings based on the content of the original XML file from which it was created. Translators do not alter the POT. The POT should only be changed after changes are made to the original XML file from which it is derived.

PO strings fall into three categories:

1. **translated**, meaning the string has been handled by a translator and its source has not changed since then
2. **fuzzy**, meaning the string has changed since it was last handled by a translator
3. **untranslated**, meaning no translation has yet been provided for this string, or it is brand new

6.2. Introducing Transifex.net

The Fedora Project uses Transifex, an open source translation platform, to power the translation of software and documentation. All translations are hosted at [Transifex.net](https://www.transifex.net/), which provides a web application allowing translators to write, submit, and manage their translations. Document translations are then pulled into the document’s Git repository, where they can be built and published to [Fedora Documentation website](https://docs.fedoraproject.org).

Documentation maintainers are responsible for the following:

- pushing changes to their guide or other document to Transifex.net
- pulling updated translations back to their git repository
- publishing the translations to [http://docs.fedoraproject.org](http://docs.fedoraproject.org)

The following sections explain how to set up Transifex for use with your document.

---

1. https://www.transifex.net/
2. https://docs.fedoraproject.org
3. https://docs.fedoraproject.org
6.3. Creating an Account on Transifex.net
Go to Transifex.net and click on Register in the top right hand corner to create an account.

6.4. Installing the Transifex Client
To use Transifex with your Git repository, you need the transifex-client package installed from the Fedora repositories.

6.5. Configuring .transifexrc
Edit the file ~/.transifexrc, adding your Transifex.net username and password in the following format:

```
[https://www.transifex.net]
hostname = https://www.transifex.net
username = username
password = password
token =
```

6.6. Initializing Transifex in Your Git Repository
Go to the Git repository of your document and change to the current Fedora release branch. Alternatively, you can create a new branch specifically for translations by branching the release branch, and then periodically merge updated content from the release branch to the branch with translations. After choosing the approach that is appropriate for you and changing to the branch, enter the following commands, replacing url with the URL of your document’s project page on Transifex.net:

```
tax init
tax set --auto-remote url
```

This will create a .tx folder in your repository.

I cannot find my document on Transifex.net!

Most Fedora documents already have a project page on Transifex.net, but if your document is new you may need to set one up. Ask on the Fedora Docs Project mailing list for help with how to do this.

6.7. Mapping Language Codes
If you have translation folders already in your repository, you may need to map the language codes to the ones used by Transifex.net, if they are different.

To do this, open the file .tx/config. You should see a [main] section which allows you to set configurations for the whole document, followed by sections for the individual files of which your

---

4 https://www.transifex.net/
document is comprised (resources, in Transifex terminology). In the [main] section, edit the lang_map line to map the language codes in your repository to the ones used on Transifex.net. For example, if your repository uses ro-RO for Romanian and bg-BG for Bulgarian, but Transifex.net uses ro and bg respectively, the lang_map line should look like this:

lang_map = ro:ro-RO,bg:bg-BG

The language code used in Transifex.net comes first, followed by the language code used by the folder in your repository. Multiple language mappings are given as a comma-separated list.

**Finding currently active Translation Teams**

To find out what Translation Teams currently have translations for your document, and to see what language code they use, click on All Resources under Project Releases on your document’s Transifex.net project page. This will bring up a list of all currently active Translation Teams, with the name of each language followed by its language code in parentheses. You can also see how much of your document each team has translated, and the details of the last submitted translation.

For your .tx/config file you can simply reuse the following language code mappings, which are utilized by a number of Fedora guides that are tracked by Transifex.net:


### 6.8. Editing File Filters

By default, transifex-client downloads translations and stores them in a translations folder in your repository. For Fedora documentation, transifex-client must be configured to download the translations to the root folder of your repository, with each language having its own folder, so that Publican can find them.

For each resource listed in the .tx/config file, edit the file_filter line to give the correct location:

file_filter = <lang>/resource.po

Replace resource with the name of the resource, which is the second part of the section title shown directly above. For example:

[fedora-release-notes.Amateur_Radio]
file_filter = <lang>/Amateur_Radio.po
source_file = pot/Amateur_Radio.pot
Congratulations, your document repository is now configured for Transifex!

### 6.9. Updating POT Files

Run the following command to update all POT files in your document's repository:

```
publican update_pot
```

After running the command, you can proceed with pushing transitions to Transifex.net. Refer to Section 6.10, “Pushing and Pulling Translations” for more information.

### 6.10. Pushing and Pulling Translations

Whenever a change is made to your document, run the `publican update_pot` command. After that, run the following command to push the changes to Transifex.net so that translators can translate them:

```
tx push -s
```

If you need to edit a `.po` file locally, you can push the changes back to Transifex.net:

```
tx push -t
```

To pull updated translations from Transifex.net:

```
tx pull -a
```

You can use the `-l` option to specify specific languages to pull, for example if you know that a particular language has new translations available.

Refer to the Fedora Wiki page Publishing a document with Publican for information on how to publish translated documentation to `docs.fedoraproject.org`.

#### Translation notifications

Make sure you select the Watch button on your document’s Transifex.net project page to receive email notifications when new translations are available. The Watch button is located below the document's title.

### 6.11. Branching and Updating Translations Before a New Release

First, change to the branch intended for the previous Fedora release. Then create a new branch for the upcoming release and change to the branch:

---

5 [https://fedoraproject.org/wiki/Publishing_a_document_with_Publican](https://fedoraproject.org/wiki/Publishing_a_document_with_Publican)
6 [https://docs.fedoraproject.org](https://docs.fedoraproject.org)
Assuming that the content development for the upcoming release happened in the master branch, which is different from the release branch, you should merge the updates from the master branch:

```
git merge master
```

After successful merging, check that the resource configuration in the `.tx/config` file is still up-to-date and applies to the current documentation. Especially pay attention to changed, moved or removed chapters in your document. If you need to remove an unused resource, run the following command:

```
tx delete -r documentation.resource
```

For example, if you want to remove the resource Amateur_Radio from the Fedora Release Notes, execute the following command:

```
tx delete -r fedora-release-notes.Amateur_Radio
```

### Use the development version

As of September 2011, you need to have the development version of `transifex-client` installed to use the `tx delete` command. Refer to [http://help.transifex.net/user-guide/client/devel.html#development-version](http://help.transifex.net/user-guide/client/devel.html#development-version) for instructions on how to get the development version.

When you are done with all necessary changes, follow the usual procedure with updating POT files and pushing and pulling translations that is described in Section 6.9, “Updating POT Files” and Section 6.10, “Pushing and Pulling Translations”.

### 6.12. Additional Resources

Setting up and using Transifex.net is also explained on Fedora Wiki, at [https://fedoraproject.org/wiki/Setting_up_a_document_with_Transifex](https://fedoraproject.org/wiki/Setting_up_a_document_with_Transifex).

### Getting more help

Transifex.net has its own comprehensive documentation for `transifex-client` at [http://help.transifex.net/user-guide/client/index.html](http://help.transifex.net/user-guide/client/index.html). You can also post questions to the Docs Project mailing list or ask in the `#fedora-docs` IRC channel if you get stuck. The L10N Guide has lots of useful information on using Transifex from a translator’s perspective: [https://fedoraproject.org/wiki/L10N/Guide](https://fedoraproject.org/wiki/L10N/Guide).
Style Guide

Your writing should be styled consistently and concisely. This section outlines style conventions to be used when writing documentation.

7.1. General tips

Be direct. Embellishments used in spoken language do not add value to a technical document. Be sparing with qualifiers like "usually," "should," and "often." The progression of the document will be apparent. Avoid narrative language such as "now that we have configured with the file, we are going to issue a command."

Be concise. Don't use bromides and frivolous clauses.

Be professional. Keep your writing relevant and impersonal. Don't refer to the writer in the first person with "I," "we," or "us," or the reader in the third person singular. Using "you" is acceptable when giving instructions, but should be avoided in any statement of fact. "You" is preferred to the gender-specific "he" or "she," the awkward "he or she," the ugly "s/he," and the excessively dry "one."

Be accurate. Use clear, contextually appropriate language.

Stay in scope. Your tips should adequately cover a stock installation of Fedora, in the present version. Consider a separate section or referring to another document if elaboration is required. Excessive asides will dilute the value of your work and create redundancies.

Proofread. Review for spelling mistakes and grammar abuse. Remove contractions, which cause translation problems.

7.2. Abbreviations, acronyms, and initialisms

When used appropriately, abbreviations, acronyms, and initialisms (for example: abbr., NASA, and UTC) improve the flow of reading by providing brevity. The first use of the represented text should be the full version, with the appropriate shortening in subsequent uses. If the use of the shortened form of a word is ambiguous in context, rewrite or use the long form.

7.3. Pictures and screen captures

The use of pictures and screen captures should be avoided whenever possible. Although they can be of aid to users, trivial differences in appearance (e.g. desktop customizations) can be distracting or confusing. In addition, the text that appears in an image is not easily translatable, so translators must generate their own image. If you are tempted to use an image, consider improving your text to better describe the instructions.

7.4. Software

Software that appears in any official Fedora documentation must adhere to these guidelines:

Software must be available in the official Fedora software repositories for the release of Fedora to which the documentation applies.

Software installation must be done via Fedora software management tools, preferably yum. If you document using GUI tools for package installation and management, cover the default application before others, and cover it consistently.

Software must not require user compilation from source, or rebuilding from source RPMs. Guides must not encourage reconfiguration of file system resources, security contexts, or any other resources from a software package that conflict with the intention of the maintainer for that package.
Appendix A. Revision History

Revision 0.3-0  Mon May 28 2012  Ben Cotton bcotton@funnelfiasco.com
Added a publican troubleshooting note.
Updated the lang_map.

Revision 0.2-0  Thu Sep 15 2011  Petr Kovář pkovar@redhat.com
Moved Translations instructions from Fedora Wiki to Translations chapter.

Revision 0.1-0  Mon Jun 13 2011  Eric Christensen sparks@fedoraproject.org
Moved Publican instructions from Fedora Wiki to Publican chapter.

Revision 0.0-0  Sun Jun 12 2011  Jared Smith jsmith@fedoraproject.org
Initial creation of book by publican
Created initial framework
Index

F
feedback
  contact information for this manual, vii

I
IRC (Internet Relay Chat)
  communicating with the Docs team, 1