Creating Manpages with groff

A Call for New Manpages!

Do have difficulty remembering command syntax? It often pays to check the manpage. In this issue of Command Line, we look at how to create your own manpages, and how to convert manpages to other formats.

BY HEIKE JURZIK

Good documentation is almost as important as good programming. It even makes sense to document the major functions in the most trivial of your own scripts, to save you from racking your brains later when you need to change something. Manpages give users tips on usage and details on command syntax options. Some manpages also include examples or references to related tools.

Users can read manpages in a terminal window, and they can convert a manpage quickly and easily to another format such as HTML, PostScript, or PDF. Manual pages are traditionally created using the text formating tool groff. The first version of this program appeared on legacy Unix systems, where it was known as roff (= "run off"). Later developments of the document formatter were called nroff and troff; groff is the GNU version for the current crop of Linux systems. This article shows how to use groff to write a manpage and how to convert that manpage into other formats.

A "man" for All Seasons

The manpages on a Linux system are stored below */usr/share/man* – this is where the subdirectories *man1* through *man9* are located. The numbers *1* through *9* tell you the topic to which the manpage has been assigned. For example, section *1* contains descriptions for user commands; the manpages in *5* describe the formats of important files, and *8* has commands for the system administrator.

The KDE Help Center below *Linux Documentation / Manpages* gives users a useful overview and short descriptions of the individual sections (see Figure 1).

It is not normally necessary to specify the section when entering the *man* command, however, if there are different manpages in different sections, you have to stipulate which manpage you mean. For example, to read the manpage for the user command *printf*, type

```
man 1 printf
```

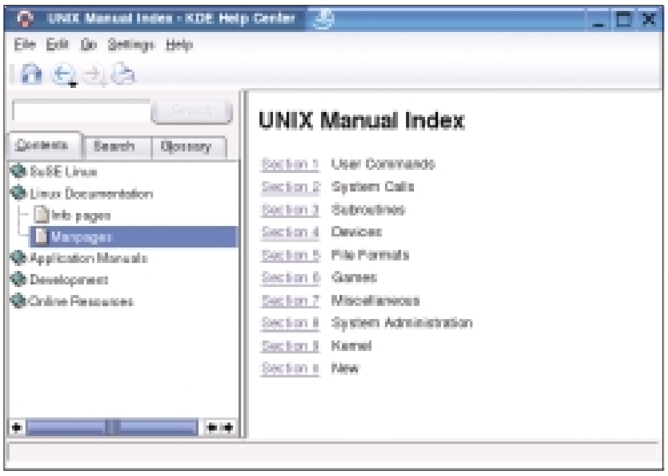
at the prompt. But if you are interested in the C library function in section 3, you need to the following instead:

```
man 3 printf
```

Under KDE, Konquerer tells you if different sets of manpages exist. In the address line, simply type *man:* followed by the command you are interested in. If there is only one manpage, Konqueror will open the page, but if there are several options, you can select the one you are interested in (Figure 2).

Structure of a Manpage

Manpages are typically divided into several chapters – at the top you see the command name and the section to which the command has been assigned. The *NAME* section of the manpage gives you a short description of the command; *SYNOPSIS* shows the command with its full range of arguments, and *DESCRIPTION* describes how the command works.


UNIX Manual Index - KDE Help Center

UNIX Manual Index

Section 1	User Commands
Section 2	System Calls
Section 3	Subroutines
Section 4	Devices
Section 5	File Formats
Section 6	Games
Section 7	Miscellaneous
Section 8	System Administration
Section 9	Kernel
Section n	New

Figure 1: The KDE Help Center sorts manpages by topic.

Table 1: Formatting in groff

.TH	Title/Headline of manpage
.SH	Chapter or section heading
.SS	Secondary heading
.PP	New paragraph / new line
.HP	Start of indented paragraph
.RE	End of indented paragraph
.IP	Enumerated or bullet list; the line following this tag needs either a letter, a bullet (*\(bu*, or a dash (*\(em*); the third line has the definition.
.TP	Plain list; the line following the formatting sequence has the identifiers; the third a description.
."	Comment
\fB	Enables ("bold") format for the following text
\fI	Enables ("italic") format for the following text
\fR	Enables serifs for the following text ("roman")
\fP	Switches back to the previous typeface

Then come the details, followed by a complete list of parameters. Some manpages have *EXAMPLES* of practical uses for the command. At the end of the manpage, you might find references to known *BUGS*, the *AUTHOR* of the manpage, and related programs (*SEE ALSO*).

Format, Format

Many manpages use the *groff* format. groff uses a text markup approach similar to HTML, SGML, or layout systems such as LaTeX. Special formatting commands allow you to describe the appearance or structure of the text. You can then convert *groff* formatted documents into formatted ASCII text, HTML pages, or even PostScript files.

Many *groff* formatting commands use lines that start with a dot followed by statements that describe the structure of the manpage. For example *.SH* indicates a chapter heading in a manpage, and *.PP* stands for a new line or new paragraph. You can also use strings to describe elements within the body text (you do not need to use a new line): for example *\fB* switches the "bold" tag on, and *\fP* switches back to the previous typeface. Table 1 gives you an overview of the major formatting commands, but you might also like to check out the manpage (*groff_man(7)*).

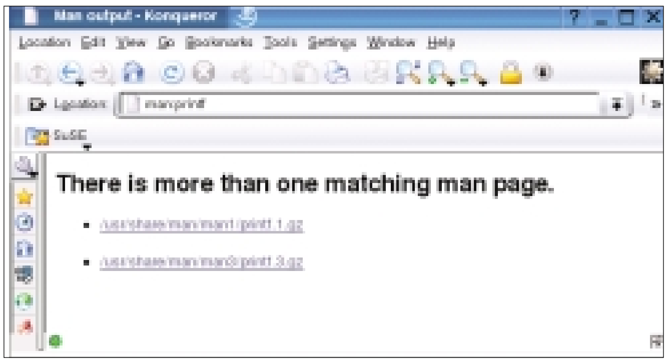
Creating a Basic Structure

Learning by doing is probably the best way to get to know groff. So let's write our own manpage. All you need to do that is a simple text editor. Create a new file, making sure you use *.1* as the suffix if this is a user command that will reside in the manpage section for user commands. The first line of a manpage traditionally has the title, the manpage section, and the last edit date; so add the following:

```
.TH CHICKEN 1 "22 January 2005"
```

The next step is to create the chapters. It is up to you to decide which of the sections we discussed earlier you will be using – this is just a convention, but not mandatory. You can define your own sections if you like. The individual sections start with the *.SH* command followed by the section name, and they need to be in a line on their own:

```
.SH NAME
.SH SYNOPSIS
.SH DESCRIPTION
.SH SEE ALSO
.SH BUGS
.SH AUTHOR
```


Man output - Konqueror

There is more than one matching man page.

- /usr/share/man/man1/printf.1.gz
- /usr/share/man/man3/printf.3.gz

Figure 2: Clicking your way to the right manpage with Konqueror.

The *DESCRIPTION* should have an *Options* subsection with the individual command parameters. Let's add a structural element:

```
.SH DESCRIPTION
.SS Options
...
```

More Content Please!

Now all you need to do is to fill the empty structure with content. To allow commands such as *apropos* or *man -k* to search your manpage for keywords, you might like to add a third line (following the *.SH NAME* entry), with the name and a short description of the command separated by a dash. Make sure you escape the dash with a backslash:

```
.SH NAME
chicken \- The sensational ⮐
chicken and egg script
```

If you cannot remember how a command is spelled, you can use *apropos <searchkey>* or *man -k <searchkey>* to list the commands that contain the search key. Manpage names and short descriptions from the third line of manpages are stored in a database – most systems use a daily cron job to update the entries. Alternatively, admins can run the *mandb* command after creating a new manpage.

The *SYNOPSIS* section has a short description of the command syntax with a list of options. The program name should be in bold type (**\fB...\fP**); parameters should be in italics (\f*I...*\f*P*), and optional parameters should be in square brackets:

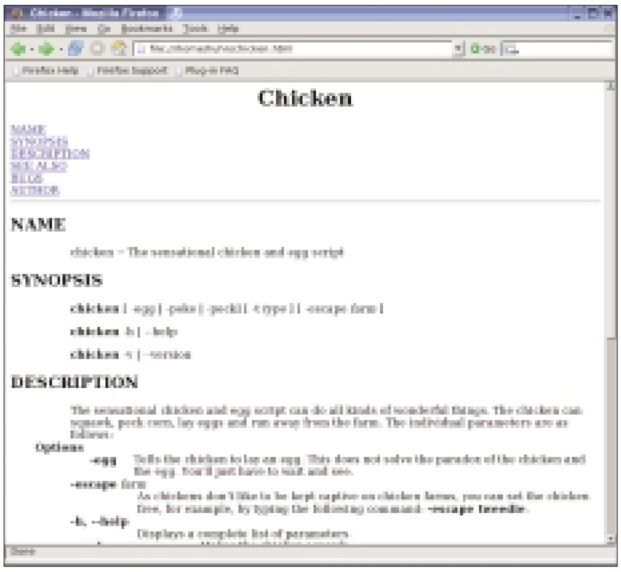
```
.SH SYNOPSIS
\fBchicken\fP [ -egg | -poke | -peck ] [ -t \fItype\fP ] [ -escape \fIfarm\fP ]
```

This is followed by a new line, a new paragraph – helping to keep the manpage readable – and a note on how to call for help. You need another new line for the version number entry:

```
.PP
\fBchicken\fP -h | --help
.PP
\fBchicken\fP -v | --version
```

In the *DESCRIPTION* section, you can add a body text description of the command. Use the formatting strings mentioned earlier to put commands, file names, and options in italics or bold type; this all helps to improve readability.

In the *Options* subsection, create a list of the individual parameters and add an explanation for each one. Use the list types described in Table 1 to do so, for example, for a plain list, you need *.TP* in the first line, the parameter in the second line, and the explanation in the third line. Follow the same pattern for the remaining sections. Listing 1 shows you the source code of the *chicken* manpage.


Chicken

NAME
SYNOPSIS
DESCRIPTION
SEE ALSO
BUGS
AUTHOR

NAME

chicken – The sensational chicken and egg script

SYNOPSIS

chicken [-egg | -poke | -peck] [-t type] [-escape farm]

chicken -h | --help

chicken -v | --version

DESCRIPTION

The sensational chicken and egg script can do all kinds of wonderful things. The chicken can squawk, peck corn, lay eggs and run away from the farm. The individual parameters are as follows:

Options

-egg Tells the chicken to lay an egg. This does not solve the paradox of the chicken and the egg. You'll just have to wait and see.

-escape farm
As chickens don't like to be kept captive on chicken farms, you can set the chicken free, for example, by typing the following command: **-escape tweedie**.

-h, --help
Displays a complete list of parameters.

Figure 3: You can use the command line to convert the manpage into a website with a single command.

Flexible

You can use the command line to quickly test whether your manpage looks like you intended it to look. Use the manpage macros to convert your manpage into ASCII format.

To stop the output from scrolling off the screen, pipe the output to the *less* pager:

```
groff -man -Tascii chicken.1 | less
```

If you prefer to generate an HTML or PostScript file instead, specify your preferred format for the *-T* option. Of course you will want to redirect the output into a file rather than messing up the screen. The following should do the trick:

Listing 1: Manpage for *chicken* (1)

```
.TH Chicken 1 "22 January 2005"
.SH NAME
chicken \- The sensational chicken and egg script
.SH SYNOPSIS
\fBchicken\fP [ -egg | -poke | -peck] [ -t \fItype\fP ] [ -escape
\fIfarm\fP ]
.PP
\fBchicken\fP -h | --help
.PP
\fBchicken\fP -v | --version
.SH DESCRIPTION
The sensational chicken and egg script can do all kinds of wonderful things. The chicken can squawk, peck corn, lay eggs, and run away from the farm. The individual parameters are as follows:
.SS Options
.TP
\fB-egg\fP
tells the chicken to lay an egg. This does not solve the paradox of the chicken and the egg. You'll just have to wait and see.
.TP
\fB-escape \fIfarm\fP
As chickens don't like to be kept captive on chicken farms, you can set the chicken free, for example, by typing the following command: \fB-escape
tweedie\fP.
.TP
\fB-h, --help\fP
displays a complete list of parameters.
.TP
\fB-poke\fP
makes the chicken squawk.
.TP
\fB-peck\fP
tells the chicken to eat something.
.TP
\fB-t \fItype\fP
defines the chicken type:
.IP \(bu
\fIfree\fP free range
.IP \(bu
\fIdeep\fP deep litter
.IP \(bu
\fIbattery\fP battery
.TP
\fB-v, --version\fP
displays the version number of the script.
.SH SEE ALSO
The Internet has lots of interesting pages on chickens. The following manpages have useful tips on creating your own manpages:

\fBgroff_man\fP(7), \fBgroff\fP(1), \fBman\fP(7)
.SH BUGS
None known :)
.SH AUTHOR
Petronella, the big boss chicken
```

```
groff -man -Thtml ➘
chicken.1 > chicken.➘
html
```

or

```
groff -man -Tps chic➘
ken.1 > chicken.ps
```

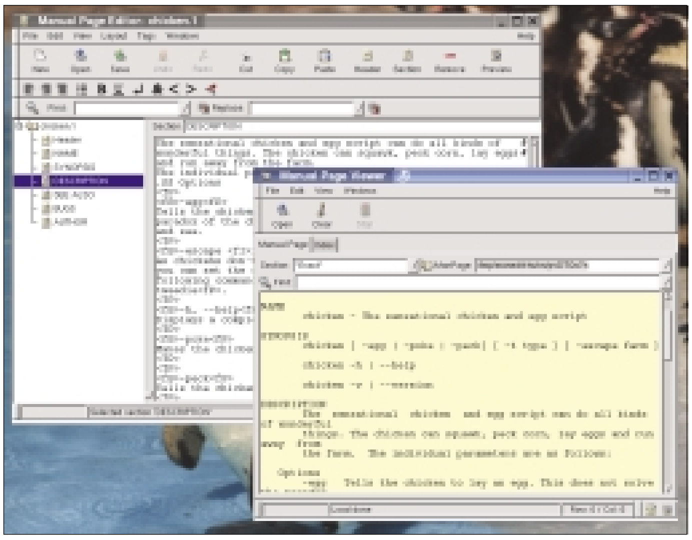
On Location

You may be wondering what you need to do with your manpage to allow other system users to type *man chicken* and read it.

It's simple: working as the administrator, copy your *chicken.1* file to the right place:

```
cp chicken.1 ➘
/usr/share/man/man1/
```

Now you can either wait for the daily manpage database update, or, working as the administrator, type



```
NAME
       chicken - The sensational chicken and egg script
SYNOPSIS
       chicken [ -egg | -poke | -peck] [ -t type ] [ -escape farm ]
       chicken -h | --help
       chicken -v | --version
DESCRIPTION
       The sensational chicken and egg script can do all kinds
of wonderful
       things. The chicken can squawk, peck corn, lay eggs and run
away from
       the farm. The individual parameters are as follows:
  Options
       -egg   Tells the chicken to lay an egg. This does not solve
```

Figure 4: Click for manpages with ManEdit.

```
mandb
```

If your page has syntax errors, the *mandb* will let you know. If everything works, you should see something like:

```
...
4 man subdirectories contained ➘
newer manual pages.
27 manual pages were ➘
added.
0 stray cats were ➘
added.
0 old database ➘
entries were purged.
```

Text Editor or GUI?

If you do not feel like memorizing all those formatting commands, you might prefer to use the GUI-based ManEdit (*http://wolfpack.twu.net/ManEdit/*) tool. Syntax highlighting, preconfigured tags, and a preview function make writing documentation child's play (Figure 4).

One thing is for sure. Even if you opt to use ManEdit, you will still benefit from having good background knowledge of manpages – being familiar with the structure, and knowing how groff works, allows for more effective use of the GUI editor. ■

Advertisment