Linux Man Page Howto

Jens Schweikhardt

howto@schweikhardt.net

Vertaald door: Ellen Bokhorst

bokkie@nl.linux.org

$Id: Man-Page-NL.sgml,v 1.1 2003/05/13 11:34:22 bokkie Exp $

In deze HOWTO wordt uitgelegd wat je in gedachten moet houden wanneer je on-line documentatie gaat schrijven, een zogenoemde manpage, die je toegankelijk wilt maken via de opdracht *man*(1). In dit document wordt naar een manual entry simpelweg gerefereerd als een manpage, ongeacht de werkelijke lengte en zonder sexistische bedoelingen.

Inhoudsopgave

1. Een paar gedachten over documentatie

Waarom schrijven we documentatie? Domme vraag. Omdat we willen dat anderen ons programma, onze bibliotheekfunctie of wat we dan ook hebben geschreven en beschikbaar hebben gemaakt kunnen gebruiken. Maar het schrijven van de documentatie is nog niet alles:

- Documentatie moet toegankelijk zijn. Als het op een of andere niet standaard lokatie is verborgen waar de aan documentatie gerelateerde tools het niet kunnen vinden, hoe kan het dan zijn doel dienen?
- Documentatie moet betrouwbaar en accuraat zijn. Er is niets ergerlijker dan het niet met elkaar overeenkomen van het functioneren van het programma en de beschrijving van de documentatie. Gebruikers zullen je vervloeken, je hatelijke mail sturen en je werk de prullenbak in werpen, met de vastbesloten bedoeling nooit meer iets geschreven door die stommeling te installeren.

De historische en welbekende wijze om onder UNIX documentatie te benaderen is via de opdracht man(1). In deze HOWTO is beschreven wat je moet doen om een manpage te schrijven die correct door de aan documentatie gerelateerde tools wordt verwerkt. De belangrijkste tools hiervan zijn `man`(1), `xman`(1x), `apropos`(1), `makewhatis`(8) en `catman`(8). De betrouwbaarheid en accuraatheid van de informatie is natuurlijk geheel aan jou. Maar zelfs dit in aanmerking genomen zul je hieronder een aantal ideeën aantreffen die je helpen een aantal valkuilen te voorkomen.

2. Hoe worden manpages benaderd?

Je moet bekend zijn met het precieze mechanisme voor het benaderen van manpages om aan je manpage de juiste naam toe te kennen en het op de juiste lokatie te installeren. Elke manpage moet in een specifieke sectie worden gecatalogiseerd, welke wordt aangeduid door een enkel teken. De meest gebruikelijke secties en de voor mensen leesbare namen zijn onder Linux:

```
Sectie  De voor mensen leesbare naam
   1    Gebruikersopdrachten die door iedereen kunnen worden opgestart
   2    Systeemaanroepen, d.w.z. functies waarin voorzien door de kernel.
   3    Subroutines, d.w.z. libraryfuncties.
   4    Devices, d.w.z., speciale bestanden in de directory /dev.
   5    Beschrijvingen van bestandsformaten, b.v. /etc/passwd.
   6    Spellen, spreekt voor zich.
   7    Diversen, b.v. macropackages, conventies.
   8    Systeembeheertools die alleen root kan uitvoeren.
   9    Een andere (Linux specifieke) plaats voor documentatie over kernelroutines.
   n    (Afgekeurd) Nieuwe documentatie, die kan worden verplaatst naar een
        beter geschikte sectie.
   o    (Afgekeurd) Oude documentatie, die voor een gepaste periode kan worden
        behouden.
   l    (Afgekeurd) Lokale documentatie refererend naar dit specifieke systeem.
```

De naam van het bronbestand van de manpage (de invoer voor het opmaaksysteem) is de naam van de opdracht, functie of bestandsnaam, gevolgd door een punt, gevolgd door het teken van de sectie. Als je documentatie schrijft over de opmaak van het bestand passwd' dan noem je het bronbestand 'passwd.5'. Hier hebben we ook een voorbeeld van een bestandsnaam die hetzelfde is als een opdrachtnaam. Het kan zijn dat er zelfs een library subroutine is met de naam passwd. De gebruikelijke wijze om deze dubbelzinnigheden op te lossen is het onderverdelen in secties: De opdrachtbeschrijving is te vinden in het bestand 'passwd.1' en de hypothetische library subroutine in 'passwd.3'.

Soms worden extra tekens toegevoegd en krijg je bijvoorbeeld een bestandsnaam als 'xterm.1x' of 'wish.1tk'. De bedoeling hiervan is aan te geven dat dit documentatie is voor respectievelijk een X Window programma of een Tk toepassing. Een aantal manual browsers is in staat gebruik te maken van deze aanvullende informatie. Bijvoorbeeld xman zal 'xterm(x)' en 'wish(tk)' gebruiken in de lijst met beschikbare documentatie.

Maak alsjeblieft geen gebruik van de secties n, o en l; volgens de File System Standard zijn deze secties afgekeurd. Houd je aan de numerieke secties. Wees je bewust van naamsaanvaringen met bestaande programma's, functies of bestandsnamen. Het is beslist geen goed idee om nog een andere editor te schrijven en het ed, sed (voor smart ed) of red (voor Rocky's ed) te noemen. Door ervoor te zorgen dat de naam van je programma uniek is, voorkom je dat iemand jouw programma uitvoert en de manpage van iemand anders leest, of andersom.

Nu weten we in ieder geval welke naam we ons bestand moeten geven. De volgende beslissing die moet worden genomen is de directory waarin het uiteindelijk zal worden geïnstalleerd (stel wanneer de gebruiker ‘`make install`’ toepast op je package.) Onder Linux staan alle manpages onder directory’s die zijn weergegeven in de omgevingsvariabele MANPATH. De aan documentatie gerelateerde tools gebruiken MANPATH op dezelfde wijze als de shell PATH gebruikt om uitvoerbare bestanden te lokaliseren. In feite heeft MANPATH hetzelfde formaat als PATH. Elk bevat een door dubbele punten gescheiden lijst met directory’s (met als uitzondering dat MANPATH geen lege velden en relatieve padnamen toestaat, het maakt alleen gebruik van absolute namen). Als MANPATH niet is ingesteld of geëxporteerd, dan zal een standaardwaarde worden gebruikt waarin op z’n minst de directory /usr/man is opgenomen. Voor het versnellen van het zoeken en het klein houden van directory’s bevatten de directory’s gespecificeerd door MANPATH (de zogenoemde basisdirectory’s) een boel subdirectory’s met de naam ‘man<s>’ waarbij <s> staat voor de sectietoekenning van één teken geïntroduceerd in bovenstaande tabel. Het kan zijn dat niet alle secties worden voorgesteld door een subdirectory omdat er simpelweg geen reden is om een lege ‘mano’ subdirectory bij te houden. Er kunnen echter directory’s voorkomen met de namen ‘cat<s>’, ‘dvi<s>’ en ‘ps<s>’ waarin documentatie wordt bewaard die klaar is voor de weergave of om af te drukken. Hierover later meer. Het enige andere bestand in een basisdirectory zou een bestand met de naam ‘whatis’ mogen zijn. Het doel en de aanmaak van dit bestand zal ook in paragraaf 12) worden beschreven. De veiligste wijze om een manpage voor sectie <s> op de juiste lokatie te hebben geïnstalleerd, is door het te plaatsen in de directory /usr/man/man<s>. Een goede `Makefile`, echter, zal de mogelijkheid bieden dat de gebruiker een basisdirectory kan kiezen, door middel van een `make` variabele, zoals bijvoorbeeld MANDIR. De meeste GNU-packages kunnen worden geconfigureerd met de optie `--prefix=/wat/dan-ook`. De manpages zullen dan onder de basisdirectory /wat/dan-ook/man worden geïnstalleerd. Ik raad je aan in iets vergelijkbaars te voorzien.

Met de komst van de Linux File System Standard (FS-Stnd), werd ’t allemaal wat gecompliceerder. [Noot: de FS-Stnd schijnt te worden vervangen door de Filesystem Hierarchy Standard (http://www.pathname.com/fhs/), FHS.] De FS-Stnd 1.2 zet uiteen dat

"Voorzieningen moeten worden getroffen in de structuur van /usr/man ter ondersteuning van manpages geschreven in andere (of meerdere) talen."

Dit wordt bewerkstelligd door het introduceren van een ander directoryniveau welk onderscheid maakt tussen de verschillende talen. Nogmaals de FS-Stnd 1.2 aanhalend:

"Deze benoeming van taalsubdirectory’s van /usr/man is gebaseerd op Appendix E van de POSIX 1003.1 standaard waarin de locale identificatiestring wordt beschreven, de best geaccepteerde methode om een culturele omgeving te beschrijven. De <locale> string is: <taal>[_<gebied>][.<tekenset>][,<versie>]"

(Zie de FS-Stnd voor een paar algemene <locale> strings.) Overeenkomstig deze richtlijnen staan onze manpages onder /usr/man/<locale>/man[1-9lno]. De opgemaakte versies staan dan natuurlijk in /usr/man/<locale>/cat[1-9lno], anders zouden we ze alleen voor een enkele locale kunnen leveren. Ik kan je ECHTER thans niet aanbevelen naar die structuur over te schakelen. De FS-Stnd 1.2 staat ook toe dat

"Systemen die een unieke taal en codeset voor alle manpages instellen, mogen de <locale> substring achterwege laten en alle manpages opslaan in <mandir>. Systemen met bijvoorbeeld alleen Engelstalige manpages gecodeerd in ASCII, mogen manpages (de `man[1-9]` directory’s) direct opslaan in /usr/man. (Dat is in feite de traditionele omstandigheid en regeling.) "

Ik zou niet overschakelen voordat alle tools (zoals xman, tkman, info en vele anderen die manpages inlezen) met de nieuwe structuur overweg kunnen.

3. Hoe hoort een opgemaakte manpage eruit te zien?

Laat me je een voorbeeld presenteren. Hieronder zal ik het in detail uitleggen. Als je dit in gewone tekst leest, zal het de verschillende lettertypen niet tonen (vet en schuindruk). [TEDOEN: de nadruk en schuindruk is verdwenen bij de conversie naar SGML/HTML; breng het terug.] Raadpleeg alsjeblieft de paragraaf "Wat zijn de fontconventies?" voor meer uitleg. Hier is de manpage voor het (hypothetische) `foo` programma.

```
FOO(1)                 Gebruikershandleidingen                FOO(1)


NAAM
    foo - frobnicate de bar library

SYNTAX
    foo [-bar] [-c configuratiebestand ] bestand ...

BESCHRIJVING
     foo  frobnicates de bar library door interne symbooltabellen
     aan te passen. Standaard verwerkt het alle baz segmenten en
     herschikt ze in omgekeerde tijdsvolgorde wil de xyzzy(1)
     linker ze kunnen vinden. De symdef entry wordt dan gecomprimeerd
     gebruik makend van het WBG (Whiz-Bang-Gizmo) algoritme. Alle bestanden
     worden in de opgegeven volgorde verwerkt.

OPTIES
     -b   Schrijf tijdens de verwerking geen ‘busy’ meldingen naar stdout.

     -c configuratiebestand
          Gebruik het alternatieve systeemomvattende configuratiebestand
          in plaats van /etc/foo.conf. Hiermee wordt de omgevingsvariabele
          FOOCONF overschreven.

     -a   Verwerk in aanvulling op de baz segmenten ook de blurfl headers.

     -r   Recursieve modus. Functioneert zo snel als het licht ten koste
          van een megabyte aan virtueel geheugen.

BESTANDEN
     /etc/foo.conf
          Het systeemomvattende configuratiebestand. Zie foo(5) voor
          meer details.
     ~/.foorc
          Configuratiebestand per gebruiker. Zie foo(5) voor meer
          details.

OMGEVING
     FOOCONF
          Als het een waarde bevat de volledige padnaam voor een
          alternatief systeemomvattend foo.conf. Wordt overschreven
          door de optie -c.

DIAGNOSE
     De volgende diagnoses kunnen op stderr worden uitgevoerd:

     Onjuist magisch nummer.
          Het invoerbestand lijkt niet op een archiefbestand.
     baz segmenten in oude stijl.
          foo kan alleen omgaan met baz segmenten in de nieuwe stijl. COBOL
          objectlibrary’s worden in deze versie niet ondersteund.

BUGS
     De opdrachtnaam zou zorgvuldiger moeten zijn gekozen om zijn
     doel weer te geven.
```

```
AUTEUR
     Jens Schweikhardt <howto at schweikhardt dot net> (mailto:howto@schweikhardt.net)

ZIE OOK
     bar(1), foo(5), xyzzy(1)

Linux                Last change: MAART 1995                    2
```

Hier is de uitleg zoals ik beloofde.

De sectie NAAM

...is de enige vereiste sectie. Man pages zonder naamsectie zijn zo bruikbaar als koelkasten op de noordpool. Deze sectie heeft tevens een gestandaardiseerde opmaak bestaande uit een door komma's gescheiden lijst met programma- of functienamen, gevolgd door een koppelteken, gevolgd door een beknopte (gewoonlijk bestaande uit een éénregelige) beschrijving van de functionaliteit die het programma (of de functie of het bestand) wordt verondersteld te leveren. Door middel van `makewhatis`(8), komen de naam secties terecht in de `whatis` databasebestanden. `Makewhatis` is de reden dat de naamsectie moet voorkomen, en waarom het moet voldoen aan de hier beschreven opmaak. In de `groff` broncode moet het er uitzien als

.SH NAAM foo \- frobnicate de bar library

De \- is hier van belang. De backslash is nodig om onderscheid te kunnen maken tussen een koppelteken en een afbreekstreepje welk in de opdrachtnaam of de éénregelige beschrijving voor kan komen.

De sectie SYNTAX

...is bedoeld voor een beknopt overzicht van beschikbare programma-opties. Bij functies wordt in deze sectie een opsomming gegeven van corresponderende include bestanden en het prototype zodat de programmeur het type en aantal argumenten weet als ook het returntype.

De sectie BESCHRIJVING

...legt welsprekend uit waarom je reeks nullen en énen ook maar iets waard is. Hier schrijf je al je kennis neer. Dit is je reputatie. Win de bewondering van andere programmeurs en gebruikers door van deze sectie de bron met betrouwbare en gedetailleerde informatie te maken. Geef uitleg over waar de argumenten voor dienen, de bestandsopmaak, welke algoritmen het vuile werk opknappen.

De sectie OPTIES

...geeft een beschrijving van hoe elke optie het functioneren van het programma beïnvloedt. Dat wist je al, nietwaar?

De sectie BESTANDEN

...een opsomming van bestanden die het programma of de functie gebruikt. Hier staan bijvoorbeeld configuratiebestanden, opstartbestanden en bestanden waar het programma op directe wijze bewerkingen op uitvoert. Het is een goed plan om de volledige padnaam van deze bestanden te geven en ervoor te zorgen dat het installatieproces het directory gedeelte aanpast overeenkomstig de voorkeuren van de gebruiker: de `groff` manpages hebben het standaardvoorvoegsel /usr/local, dus refereren ze standaard naar /usr/local/lib/groff/* . Als je echter tijdens de installatie gebruik maakt van 'make prefix=/opt/gnu' dan veranderen de verwijzingen in de manpage in /opt/gnu/lib/groff/*

De sectie OMGEVING

...geeft een opsomming van alle omgevingsvariabelen die van invloed zijn op je programma of functie en vertelt uiteraard hoe. Het meest gebruikelijk is dat de waarden van variabelen padnamen, bestandsnamen of standaardopties zijn.

De sectie DIAGNOSE

...zou een overzicht moeten geven van de meest gebruikelijke foutmeldingen van je programma en hoe hier mee om te gaan. Het is niet nodig om foutmeldingen van het systeem uit te leggen (van `perror`(3)) of fatale signalen (van `psignal`(3)) aangezien die tijdens de uitvoering van elk programma plaats kunnen vinden.

De sectie FOUTEN

...zou ideaal gezien niet bestaan. Als je flink bent, kun je hier de beperkingen, bekende ongemakken en voorzieningen beschrijven die anderen als onvoorzieningen kunnen beschouwen. Hernoem het tot de TODO sectie als je niet zo flink bent ;-)

De sectie AUTEUR

...is aardig als het er is voor het geval er grove fouten in de documentatie of het functioneren van het programma voorkomen en je een foutenrapport wilt mailen.

De sectie ZIE OOK

...bestaat uit een lijst met gerelateerde manpages in alfabetische volgorde. Conventioneel is het de laatste sectie. Je bent er vrij in andere secties te bedenken als ze echt niet in één van de secties die tot dusverre zijn beschreven passen. Dus hoe precies genereerde je die manpage? Ik verwachtte die vraag, hier is de broncode:

```
.\" Verwerk dit bestand met
.\" groff -man -Tascii foo.1
.\"
.TH FOO 1 "MAART 1995" Linux "Gebruikershandleidingen"
.SH NAAM
foo \- frobnicate de bar library
.SH SYNTAX
.B foo [-bar] [-c
.I configuratiebestand
.B ]
.I bestand
.B ...
.SH BESCHRIJVING
.B foo
frobnicates de bar library door interne symbooltabelen aan te passen.
Standaard verwerkt het alle baz segmenten
en herschikt ze in omgekeerde tijdsvolgorde zodat de
.BR xyzzy (1)
linker ze kan vinden. De symdef entry wordt dan gecomprimeerd
gebruik makend van het WBG (Whiz-Bang-Gizmo) algoritme.
Alle bestanden worden in de opgegeven volgorde verwerkt.
.SH OPTIES
.IP -b
Schrijf tijdens de verwerking geen ‘busy’ meldingen naar stdout.
.IP "-c configuratiebestand"
Gebruik het alternatieve systeemomvattende
.I configuratiebestand
in plaats van
.IR /etc/foo.conf .
Dit overschrijft een
.B FOOCONF
omgevingsvariabele.
.IP -a
Verwerk in aanvulling op de baz segmenten ook de blurfl headers.
.IP -r
Recursieve modus. Functioneert zo snel als het licht
ten koste van een megabyte aan virtueel geheugen.
.SH BESTANDEN
```

```
.I /etc/foo.conf
.RS
Het systeemomvattende configuratiebestand. Zie
.BR foo (5)
voor meer details.
.RE
.I ~/.foorc
.RS
Configuratiebestand per gebruiker. Zie
.BR foo (5)
voor meer details.
.SH OMGEVING
.IP FOOCONF
Als het een waarde bevat de volledige padnaam voor een
alternatief systeemomvattend
.IR foo.conf .
Wordt overschreven door de optie
.B -c .
.SH DIAGNOSE
De volgende diagnoses kunnen op stderr worden uitgevoerd:

Onjuist magisch nummer.
.RS
Het invoerbestand lijkt niet op een archiefbestand.
.RE
baz segmenten in oude stijl.
.RS
.B foo
kan alleen omgaan met baz segmenten in de nieuwe stijl. COBOL
objectlibrary's worden in deze versie niet ondersteund.
.SH BUGS
De opdrachtnaam zou zorgvuldiger moeten zijn gekozen
om zijn doel weer te geven.
.SH AUTEUR
Jens Schweikhardt <howto at schweikhardt dot net>
.SH "ZIE OOK"
.BR bar (1),
.BR foo (5),
.BR xyzzy (1)
```

4. Hoe documenteer ik verscheidene programma's/functies in een enkele manpage?

Veel programma's (`grep`, `egrep`) en functies (`printf`, `fprintf`, ...) zijn in een enkele manpage gedocumenteerd. Deze manpages zouden echter tamelijk onbruikbaar zijn als ze slechts onder één naam toegankelijk zouden zijn. We kunnen van een gebruiker niet verwachten te onthouden dat de `egrep` manpage in werkelijkheid de `grep` manpage is. Daarom is het nodig de manpage onder verschillende namen beschikbaar te stellen. Je hebt verschillende manieren om dit te bereiken:

1. maak voor elke naam identieke kopieën.
2. verbind alle manpages met elkaar met behulp van hardlinks.
3. symbolische links verwijzend naar de feitelijke manpage.

4. gebruik het `groff`'s 'source' mechanisme geleverd door de `.so` macro.

De eerste manier is vanzelfsprekend een verspilling van diskruimte. De tweede is niet aan te bevelen omdat intelligente versies van het programma `catman` veel werk kunnen besparen door het type bestand of de inhoud ervan te bekijken. (Het doel van `catman` is het formatteren van alle manpages zodat ze snel kunnen worden weergegeven.) Het derde alternatief heeft een klein nadeel: als flexibiliteit van belang is, dan moet je je er bewust van zijn dat er bestandssystemen zijn die geen symbolische links ondersteunen. Eind van het liedje is dat het 't beste is (TM) om gebruik te maken van het sourcemechanisme van `groff`. Dat doe je zo: Als je wilt dat je manpage in sectie 1 onder de namen 'foo' en 'bar' beschikbaar is, dan plaats je de manpage in foo.1 en zorg je dat bar.1 er als volgt uitziet:

```
.so man1/foo.1
```

Het is van belang het `man1/` directory gedeelte te specificeren als ook de bestandsnaam 'foo.1' omdat ten tijde dat `groff` wordt uitgevoerd door de browser de basisdirectory van de manpages de huidige werkdirectory zal zijn en `groff` interpreteert `.so` argumenten relatief aan de huidige werkdirectory.

5. Welk macropackage kan ik het beste gebruiken?

Er zijn een aantal macropackages speciaal ontworpen voor gebruik bij het schrijven van manpages. Gewoonlijk zijn ze te vinden in de groff macrodirectory /usr/lib/groff/tmac. De bestandsnamen zijn `tmac.`<iets>, waarbij <iets> het argument is aan de -m optie van groff. Groff zal `tmac.`<iets> gebruiken wanneer het de optie '`-m` <iets>' meekrijgt. Vaak wordt de spatie tussen '`-m`' en '<iets>' weggelaten, dus kunnen we zeggen '`groff -man`' wanneer we manpages met het macropackage `tmac.an` formatteren. Dat is de reden voor de vreemde naam 'tmac.an'. Afgezien van tmac.an is er nog een ander populair macropackage, genaamd `tmac.doc`, welk zijn oorsprong vindt op de Universiteit van Californië op Berkeley. Veel BSD manpages gebruiken het en het schijnt dat UCB het tot standaard heeft verheven voor documentatie. De `tmac.doc` macro's zijn veel flexibeler, maar helaas zijn er manpage browsers die ze niet zullen gebruiken en altijd `groff -man` zullen aanroepen. Bijvoorbeeld alle `xman` programma's die ik heb gezien zullen manpages die `tmac.doc` nodig hebben verknallen. Dus doe jezelf een plezier: gebruik `tmac.an` -- het gebruik van een ander macropackage wordt als schadelijk aangemerkt. `tmac.andoc` is een pseudo macropackage dat de broncode bekijkt en dan `tmac.an` of `tmac.doc` laadt. In feite zou elke manpagebrowser het moeten gebruiken, maar tot op heden, doen ze dit niet allemaal, dus is het 't beste als ons we vasthouden aan de oude `tmac.an`. Alles wat ik je van nu af aan vertel met betrekking tot macro's geldt alleen voor `tmac.an`. Als je toch gebruik wilt maken van de `tmac.doc` macro's, bekijk dan de tutorialsampler, `mdoc.samples` (http://www.freebsd.org/cgi/man.cgi?query=mdoc.samples). Een aantal distro's (is me verteld) worden ook met mdoc(7), mdoc.samples(7) en groff_man(7) geleverd.

De definitieve bron voor `troff`, waarin alle macro's worden uitgelegd is de Troff User's Manual, beschikbaar in html (http://cm.bell-labs.com/sys/doc/troff.html), PostScript (ps, 760K) (http://cm.bell-labs.com/sys/doc/troff.ps) of Portable Document Format (pdf, 240K) (http://cm.bell-labs.com/sys/doc/troff.pdf). door Jospeh F. Ossanna en Brian W. Kernighan, gereviseerd November 1992. AT&T Bell Labs heeft het voor het publiek beschikbaar gesteld. Vergeet niet de laatste geweldige homepage van W. Richard Steven (http://www.kohala.com/start/) te bekijken (beroemd om Unix Network Programming als ook de TCP/IP Illustrated trilogie), die ook een lijst heeft met Troff Resources (http://www.kohala.com/start/troff/troff.html) waaronder `tbl`, `eqn`, `pic` en andere filters.

6. Welke preprocessors kan ik gebruiken?

Groff wordt op z'n minst met drie preprocessors geleverd, `tbl`, `eqn`, en `pic` (op een aantal systemen worden ze `gtbl`, `geqn` en `gpic` genoemd.) Hun doel is preprocessor macro's en hun data om te zetten in reguliere troff invoer. `Tbl` is een preprocessor voor tabellen, `eqn` is een preprocessor voor vergelijkingen/berekeningen en `pic` is een preprocessor voor afbeeldingen. Raadpleeg alsjeblieft de manpages voor meer informatie over welke functionaliteit ze leveren. In een notedop: schrijf geen manpages waarvoor een preprocessor nodig is. Eqn zal gewoonlijk afschuwelijke uitvoer produceren voor schrijfmachine-achtige devices, helaas het type device waarop 99% van alle manpages worden bekeken (tenminste dat doe ik). Bijvoorbeeld XAllocColor.3x maakt gebruik van een paar formules met machtsverheffen.

Vanwege de aard van schrijfmachine-achtige devices zal de exponent op dezelfde regel verschijnen als de basis. N tot de macht van twee verschijnt als ‘N2’. `Tbl` kan maar beter worden vermeden omdat alle xman programma’s die ik heb gezien er op mislukken. Xman 3.1.6 maakt gebruik van de volgende opdracht om manpages te formatteren, b.v. signal(7):

```
gtbl /usr/man/man7/signal.7 | geqn | gtbl | groff -Tascii -man /tmp/xmana01760 2> /dev/null
```

wat bronnen verknalt waarin gebruik wordt gemaakt van `gtbl`, omdat de uitvoer van `gtbl` weer als invoer wordt gegeven aan `gtbl`. Het effect is een manpage zonder je tabel. Ik weet niet of het een programmeerfout of een feature is dat `gtbl` zich verslikt in zijn eigen uitvoer of dat xman een beetje slimmer zou kunnen zijn door `gtbl` niet tweemaal te gebruiken. Bovendien maakt een aantal systemen gebruik van `grog` om vast te stellen welke opties aan groff door te geven. Helaas gist grog soms verkeerd en beveelt `groff -t` aan wanneer in feite `tbl` moet worden gebruikt. We blijven in feite met twee oplossingen achter voor tabellen:

1. Maak de tabel zelf handmatig op en plaats het tussen .nf en .fi regels zodat het ongeformatteerd blijft. Je hebt op deze wijze geen vet en schuindruk, maar je tabel wordt altijd maar weer geslikt.
2. Gebruik de `tbl` macro’s die je wilt, maar distribueer de `tbl` uitvoer in plaats van de invoer. Je hebt echter te maken met de eigenaardigheid van `grog` die denkt dat een bestand met een regel beginnend met `.TS` `tbl` vereist. `Tbl` uitvoer bevat om voor mij onbekende reden nog steeds de `.TS` en `.TE`. Het schijnt dat je ze simpelweg kunt verwijderen en dat het resultaat er dan nog steeds ok uitziet. YMMV, dus test het alsjeblieft bij je manpage.

Ik moet nog een manpage te zien krijgen die de `pic` preprocessing nodig heeft. Maar ik zou het niet prettig vinden. Zoals je hierboven kunt zien, zal `xman` het niet gebruiken en `groff` zal beslist die fantastische gekkigheid op de invoer toepassen.

7. Moet ik broncode en/of reeds geformatteerde documentatie distribueren?

Laat me de voors (+) en tegens (-) geven van een paar uitgekozen mogelijkheden:

1. Alleen de broncode:+ kleiner distributiepackage.- ontoegankelijk op systemen zonder `groff`.
2. Alleen ongecomprimeerd formaat:+ zelfs toegankelijk op systemen zonder `groff`.- de gebruiker kan geen dvi of postscript bestand genereren. - verspilling van diskruimte op systemen die ook gecomprimeerde manpages hanteren.
3. Alleen gecomprimeerd formaat:+ zelfs toegankelijk op systemen zonder `groff`.- de gebruiker kan geen dvi of postscript bestand genereren. - welk compressieformaat zou je gebruiken? .Z? .z? .gz? Allemaal?
4. Broncode en ongecomprimeerd geformatteerd:+ toegankelijk zelfs op systemen zonder `groff`.- groter distributie package- een aantal systemen verwacht wellicht gecomprimeerde manpages.- overtollige informatie op systemen uitgerust met `groff`.

IMHO is het ’t beste alleen de broncode te distribueren. Het argument dat het ontoegankelijk is op systemen zonder `groff` doet er niet toe. De 500+ manpages van het Linux Documentatie Project bestaan alleen uit broncode. De manpages van XFree86 bestaan alleen uit broncode. De manpages van de FSF bestaan alleen uit broncode. In feite heb ik zelden software gezien die werd gedistribueerd met geformatteerde manpages. Als een systeembeheerder het werkelijk belangrijk vindt dat manpages toegankelijk zijn dan heeft hij ook `groff` geïnstalleerd.

8. Wat zijn de fontconventies?

Ten eerste: maak geen gebruik van directe fontoperators zoals `\fB`, `\fP` enz. Gebruik macro’s die argumenten accepteren. Op deze wijze voorkom je een gebruikelijke valkuil: het wijzigen van het font aan het einde van het woord en dat de vette tekst of schuindruk tot aan de volgende fontwijziging doorgaat. Geloof me, het gebeurt vaker dan je denkt. De `tmac.an` macro's voorzien in de volgende lettertypen:

.B Vet

.BI Vet afgewisseld met schuindruk

.BR Vet afgewisseld met Roman

.I Schuindruk

.IB Schuindruk afgewisseld met vet

.IR Schuindruk afgewisseld met Roman

.RB Roman afgewisseld met vet

.RI Roman afgewisseld met schuindruk

.SM Klein (9/10 geschaald van de reguliere grootte)

.SB Klein vet (niet klein afgewisseld met vet)

X afgewisseld door Y betekent dat de oneven argumenten grafisch worden gezet in X terwijl de even argumenten worden gezet in Y. Bijvoorbeeld

.BI "Arg 1 is Vet, " "Arg 2 is Schuindruk, " "en Vet, " "en Schuindruk."

De dubbele aanhalingstekens zijn nodig om witruimte in een argument op te kunnen nemen; zonder deze aanhalingstekens verschijnt er geen witruimte tussen de afwisselende lettertypen. In feite hoef je de macro's voor afwisselende lettertypen alleen te gebruiken in die gevallen waar je witruimte tussen lettertype wijzigingen wilt voorkomen. Tot zover wat beschikbaar is. Hieronder hoe je gebruik zou moeten maken van de verschillende lettertypen: (delen schaamteloos overgenomen uit man(7))

Alhoewel er in de UNIX-wereld veel willekeurige conventies bestaan voor manpages, definieert het bestaan van verscheidene honderden Linux specifieke manpages onze standaards: Bij functies worden de argumenten altijd gespecificeerd in schuindruk, zelfs in de SYNTAX sectie, waar de rest van de functie vet wordt afgedrukt:

.BI "mijnfunctie(int " argc ", char **" argv);

Bestandsnamen staan altijd in schuindruk, behalve in de sectie SYNTAX, waarin opgenomen bestanden vet zijn afgedrukt. Dus je zou gebruik kunnen maken van

.I /usr/include/stdio.h

en

.B #include <stdio.h>

Speciale macro's, die gewoonlijk in hoofdletters staan, worden vet afgedrukt:

.B MAXINT

Bij het nummeren van een lijst met foutcodes, staan de codes vet afgedrukt. Deze lijst maakt gewoonlijk als volgt gebruik van de macro .TP (paragraaf met hangende tag):

.TP.B EBADF.I fd is geen geldige file descriptor..TP.B EINVAL.I fd is ongeschikt als leesstof

Elke verwijzing naar een andere manpage (of naar het onderwerp van de huidige manpage) is vet afgedrukt. Als het nummer van de manpage wordt gegeven, dan is dit in roman, zonder spaties:

.BR man (7)

Acroniemen zien er het beste uit wanneer ze typografisch worden gezet in een klein lettertype. Dus is mijn aanbeveling

.SM UNIX

.SM ASCII

.SM TAB

.SM NFS

.SM LALR(1)

9. Je manpage oppoetsen

Hieronder staan een aantal richtlijnen die de betrouwbaarheid, leesbaarheid en 'vorming' van je documentatie verbeteren.

- Test voorbeelden om er zeker van te zijn dat ze werken (gebruik knippen en plakken om je shell de exacte bewoording uit de manpage te geven). Kopieer de uitvoer van je opdracht in je manpage, typ niet slechts in wat je denkt dat je programma zal afdrukken.
- Proeflees het, pas de spellingscontrole erop toe en laat iemand anders het lezen, vooral als Engels je moedertaal niet is. De HOWTO die je aan het lezen bent heeft de laatste test doorstaan (speciale dank aan Michael Miller voor een nogal heldhaftige bijdrage! Alle resterende ruwe kantjes zijn geheel mijn fout). Extra vrijwilligers zijn altijd welkom.
- Test je manpage: Produceert `groff` foutmeldingen wanneer je je manpage formatteert? Het is aardig als je de `groff` opdrachtregel in een commentaarregel plaatst. Produceert de opdracht `man`(1) foutmeldingen wanneer je `man jeprog` aanroept? Produceert het 't verwachte resultaat? Zullen `xman`(1x) en `tkman`(1tk) met je manpage overweg kunnen? XFree86 3.1 heeft xman 3.1.6 - X11R6, het zal proberen te decomprimeren met `gzip -c -d < %s > %s zcat < %s > %s`
- Zal `makewhatis`(8) de éénregelige beschrijving uit de NAAM sectie kunnen extraheren?
- Zet je manpage om in HTML opmaak met behulp van `rman` van http://polyglotman.sourceforge.net/, en bekijk het resultaat met een set webbrowsers (netscape, mozilla, opera, lynx, ...) Controleer of de kruisverwijzingen tussen je manpages als hyperlinks in de gegenereerde HTML functioneren. Als er een website bestaat voor je softwarepackage, plaats daar dan de manpages, en houd ze bijgewerkt.
- Het utility `rman` kan manpages ook omzetten naar LaTeX, RTF, SGML en andere formaten; kijk deze na als je je manpages in een boek of ander groot document wilt opnemen.
- Probeer je manpage om te zetten in HTML met behulp van `man2html`, wat al sinds man-1.4 onderdeel uitmaakt van het Linux man-package. Het `man2html` utility is een minder grootse vertaler dan `rman`, maar vrijwel elke Linux gebruiker heeft het al, dus het is het waard te zorgen dat `man2html` zich niet verslikt in je manpage.

10. Hoe krijg ik een gewoon manpage in tekstformaat zonder al die ^H^'s ?

Kijk eens naar `col`(1), omdat `col` backspacereeksen kan filteren. Gewoon voor het geval je niet zo lang kan wachten:

```
funnyprompt$ groff -t -e -mandoc -Tascii manpage.1 | col -bx > manpage.txt
```

De `-t` en `-e` switches vertellen `groff` voor te verwerken met `tbl` en `eqn`. Dit is overdreven voor manpages waar geen voorbewerking voor nodig is, maar afgezien van een paar verspilde CPU cycli kan het geen kwaad. Aan de andere kant kan het wel kwaad geen `-t` toe te passen wanneer het in feitelijk nodig is: de tabel wordt afschuwelijk opgemaakt. Je kunt er zelfs achterkomen (nou ja, gissen is een beter woord) welke opdracht nodig is om een bepaald `groff` document (niet gewoon manpages) te formatteren door het aanroepen van:

```
funnyprompt$ grog /usr/man/man7/signal.7
groff -t -man /usr/man/man7/signal.7
```

"Grog" staat voor "GROff Guess", en het doet wat het zegt--gissen. Als het perfect zou zijn, dan zouden we geen opties meer nodig hebben. Ik heb het onjuist zien gissen bij macropackages en preprocessors. Hier is een door mij geschreven klein perlscript dat de paginakoppen en voettekten kan verwijderen, je daarbij een paar pagina's besparend (en moeder natuur een boom) bij het afdrukken van lange en uitgebreide manpages. Bewaar het in een bestand met de naam `strip-headers` & chmod 755.

```
#!/usr/bin/perl -wn
#  laat het 't hele bestand in één keer inlezen:
undef $/;
#  verwijder eerste koptekst:
s/^\n*.*\n+//;
#  verwijder laatste voettekst:
s/\n+.*\n+$/\n/g;
#  verwijder pagina-overgangen:
s/\n\n+[^ \t].*\n\n+(\S+).*\1\n\n+/\n/g;
#  combineer twee of meer lege regels tot één lege regel:
s/\n{3,}/\n\n/g;
#  bekijk wat er over is...
print;
```

Je moet het als het eerste filter gebruiken na de opdracht `man` aangezien het terugvalt op het aantal newlines dat door `groff` wordt uitgevoerd. Bijvoorbeeld:

```
funnyprompt$ man bash | strip-headers | col -bx > bash.txt
```

11. Hoe krijg ik een PostScript manpage van hoge kwaliteit?

```
funnyprompt$ groff -t -e -mandoc -Tps manpage.1 > manpage.ps
```

Druk dat af of bekijk het met je favoriete PostScript printer/viewer. Zie vraag 10) voor een uitleg van de opties.

12. Hoe krijg ik 'apropos' en 'whatis' werkend?

Stel dat je je afvraagt welke compilers op je systeem zijn geïnstalleerd en hoe deze kunnen worden aangeroepen. Om deze (veelgestelde vraag) te beantwoorden, geef je op

```
funnyprompt$ apropos compiler
f77 (1) - Fortran 77 compiler
gcc (1) - GNU C and C++ compiler
pc (1) - Pascal compiler
```

`Apropos` en `whatis` worden gebruikt om snel te rapporteren welke manpage informatie bevat over een bepaald onderwerp. Beide programma's doorzoeken een aantal bestanden met de naam 'whatis' die is te vinden in elk van de basisdirectory's van de manpages. Zoals eerder uiteengezet, bevatten de whatis databasebestanden een éénregelige ingang voor elke manpage in de respectieve directorystructuur. In feite is die regel exact gelijk aan de sectie NAAM (om precies te zijn: samengevoegd op één regel waarbij het verbindingsteken is verwijderd; de sectie wordt tussen haakjes vermeld). De whatis databasebestanden worden aangemaakt met het programma `makewhatis`(8). Er zijn verscheidene versies in omloop, dus raadpleeg alsblieft de manpage om vast te stellen welke opties beschikbaar zijn. Wil `makewhatis` de NAAM secties correct kunnen extraheren dan is het van belang dat jij, de schrijver van de manpage, je houdt aan de opmaak van de NAAM sectie zoals beschreven onder vraag 3). De verschillen tussen `apropos` en `whatis` zijn simpelweg waar in de regel ze kijken, en waar ze naar zoeken. `Apropos` (wat equivalent is aan `man -k`) doorzoekt de argumentstring overal op de regel, terwijl `whatis` (equivalent aan `man -f`) een volledige opdrachtnaam alleen in dat deel voor het streepje probeert te matchen. Als consequentie zal '`whatis cc`' het rapporteren als er een `cc` manpage is en stil blijven voor `gcc`.

Correcties en suggesties welkom!

13. Kopieervoorwaarden

Copyright 1995-2001 by Jens Schweikhardt. All rights reserved.

```
"Two clause" BSD License:

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:
1. Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in the
   documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ■AS IS■ AND ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT,
INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.
```

14. Erkenningen

- Michael Miller voor het proeflezen van de volledige HOWTO (in februari 2001); Gordon Torrie voor de vele behulpzame grammaticale opmerkingen (in augustus 2001). Eventuele resterende grammaticale of stijlfouten zijn geheel mijn fout.
- S.u.S.E. (.de) (http://www.SuSE.de/) (of .com (http://www.SuSE.com/)) de enige distributeur die me gratis kopieën van hun laatste product blijven sturen, mijn werk als howto auteur erkennend.
- George B. Moody voor aanvullende suggesties hoe een manpage op te poetsen.

Laat een bericht bij me achter als je naam hier ontbreekt.

15. Changelog

- Maart 6 2001: HTML broncode doorstaat nu `weblint -pedantic`. Paragraph 6: Toegevoegd workarounds voor `tbl` screw-ups. Toegevoegd Erkenningen en Changelog. RCS Id toegevoegd.
- Augustus 9 2001: Howto geplaatst onder een twee clausule BSD licentie.

- Augustus 20 2001: Grammatica verbeterd. Gebruik een genummerde lijst voor de inhoudsopgave.
- Oktober 28 2001: Referenties naar mdoc(7), mdoc.samples(7) en groff_man(7) toegevoegd.
- April 28 2002: Grammaticacorrectie.
- April 30 2002: Werkte de link bij naar de groff_mdoc BSD tutorial.
- November 29 2002: Meer suggesties voor het oppoetsen van je manpage.
- December 15 2002: Publiceer SGML afgeleid van HTML. Verwijderde dode link naar LSM.