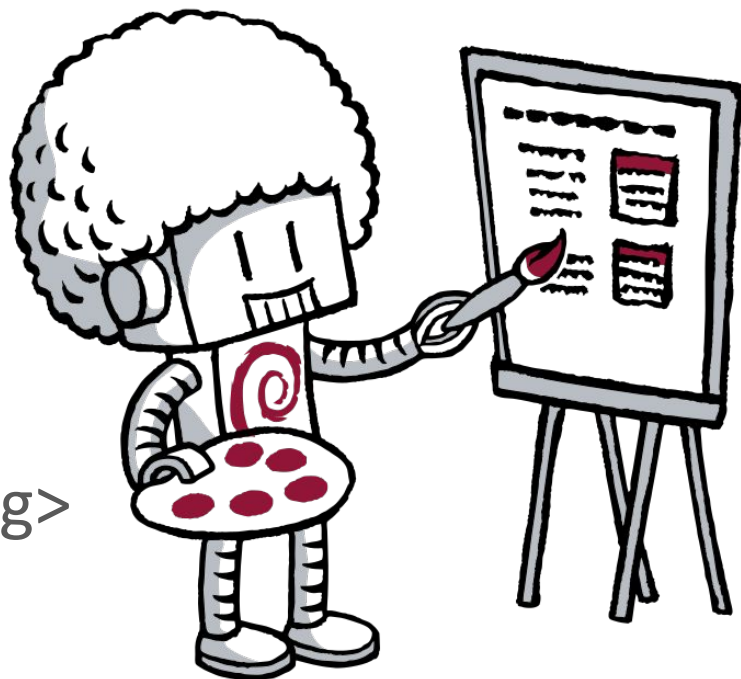


manpages.debian.org

Easterhegg 2017
Michael Stapelberg
<stapelberg@debian.org>



Agenda

- Motivation
- Übersicht
 - manpages extrahieren
 - manpages nach HTML wandeln
 - manpages ausliefern
- Laufzeitumgebung
- Fazit und Fragen

Motivation

- manpages im Browser lesen ist oft bequem
- existierende manpage-Archive generell unvollständig, alt, voll mit Werbung
- manpages.debian.org war zeitweise überlastet, generell wenig bekannt und wirkte mittlerweile recht altbacken

Kurze Demo

- für diejenigen, die manpages.debian.org noch nicht gesehen haben — Details folgen im Vortrag

Übersicht (sehr grob)

- manpages werden aus Debian extrahiert
- manpages werden nach HTML umgewandelt
- manpages werden an Endnutzer ausgeliefert

manpages extrahieren

- Debian besteht aus „source packages“, aus denen „binary packages“ gebaut werden
- während z.B. [Debian Code Search](#) nur source packages indexiert, nutzt manpages.d.o nur binary packages: viele Pakete generieren ihre manpages
- Pakete ohne `/usr/share/man` (laut `Contents`) werden übersprungen
→ nur 10G statt 50G laden beim ersten Durchlauf

manpages extrahieren (2)

- binary packages sind Architektur-abhängig, z.B. amd64 vs. i386
- manpages befinden sich in `/usr/share/man`¹, was Architektur-unabhängig² ist
- aber: manche Pakete gibt es nur für wenige Architekturen, z.B. zsnex (nur i386)
→ wir müssen den Zusammenschluss aller Architekturen verarbeiten

Dateistruktur in `/usr/share/man`

- `/usr/share/man/(<lang>/|)man<section>/<name>.<section>.gz`
 - `/usr/share/man/man1/wget.1.gz`
 - `/usr/share/man/fr.ISO8859-1/man8/iwlist.8.gz`
 - `/usr/share/man/ca@valencia/man1/deja-dup.1.gz`
- Allerlei [unsinnige Pfade in Debian](#):
 - Dateien in `man/py/man1` (von `.py`-Dateien)
 - falsches Suffix (z.B. `man1/foo.3.gz`)
 - ...

Encoding

- Default für manpages: ISO-8859-1,
aber für jedes subdirectory anders, z.B. `hu/` nutzt ISO-8859-2
...und natürlich überschreibbar via subdirectory (z.B. `fr.ISO8859-1/`)
- Trotz fehlender Kennzeichnung nutzen in der Praxis 98,8% manpages UTF-8.
- debiman wandelt beim Einlesen alles in UTF-8
→ wenn später alles UTF-8 ist, haben wir es einfach

Der .so-Mechanismus

- Manpages können andere Dateien einbinden (include),
z.B. header- oder footer-Fragmente, die vielen Manpages gemein sind
- man(1) nutzt zsoelim(1) um .so-Direktiven zu ersetzen
- Randnotiz: der Fehlerfall führt zu einer Ersetzung durch nichts, und es gibt mindestens ein Paket mit fehlerhaften .so-Direktiven

Manpages nach HTML wandeln

- erster Versuch: [grohtml](#)
→ erzeugt sehr vollständige Seiten, aber mit HTML-Tabellen und Bildern
- zweiter Versuch: [doclifter](#), wandelt manpages nach docbook
→ inkompatibel mit >50% an manpages, langsam, Encoding-Probleme
- schließlich: [mandoc](#), ein Rewrite von OpenBSD-Leuten
→ Schnell! Produziert semantisches HTML5. Ziemlich kompatibel (nicht perfekt)

mandoc: batch-Modus

- mandoc liest Dokumente von `stdin`, schreibt nach `stdout/stderr`
- Umwandeln von 470.000 manpages: 2 Stunden
 - größtenteils `fork/exec`-Overhead
 - was ist die eleganteste Art, mandoc einen batch-Modus zu verpassen?

mandoc: batch-Modus (2)

- [socketpair\(2\)](#), einen Socket an `mandocd` vererben beim starten
- über den Socket 3 file descriptors schicken als Ersatz für `stdin`, `stdout`, `stderr`
 - siehe [cmsg\(3\)](#) für Versenden von file descriptors
 - siehe [dup2\(2\)](#) für Ersetzen von file descriptors
- Implementation in [debiman](#) und [mandoc](#)

Manpages nach HTML wandeln: cross-referencing

- nach allem suchen, was wie `<name>` (`<section>`) aussieht
 - nein, es gibt kein (verbreitetes) Markup dafür
 - Formatierungs-direktiven ignorieren
- erfordert, dass die Namen aller Manpages vor dem Umwandeln bekannt sind
- selber Mechanismus für URLs. Edge-case: URLs, die cross-references enthalten

Manpages nach HTML wandeln: table of contents

- nach `<h1>`- bis `<h6>`-Elementen suchen, IDs generieren

- HTML5-IDs: at least 1 character, no spaces:

```
id := strings.Replace(text, " ", "_", -1)
```

```
u := url.URL{Fragment: id}
```

```
fmt.Println(u.String())
```

Manpages ausliefern: URLs

- /(<suite>/) (<binarypkg/>) <name> (.<section> (.<lang>))

jeder Teil (außer <name>) kann weggelassen werden

→ manpages.debian.org/i3

→ manpages.debian.org/stable/i3.1

→ manpages.debian.org/stretch/i3.1

→ manpages.debian.org/jessie/cron/crontab.5.en.html

- Perma-Links am besten ohne Sprache, damit die richtige gewählt wird

Manpages ausliefern: redirects

- [ausführlicher Mechanismus](#), um URLs zu vervollständigen
- als separater Server implementiert (`debiman-auxserver`)
 - Hauptteil der Seite kann statisch ausgeliefert werden
 - graceful degradation

Manpages ausliefern: Sprach-Erkennung

- Accept-Language HTTP-Header

z.B. `fr-CH, fr;q=0.9, en;q=0.8, de;q=0.7, *;q=0.5`

- vollständiger [Language Match](#)

z.B.: Norwegisch-Sprecher können dänische Seiten verstehen

Manpages ausliefern: opensearch.xml

- Beschreibungsdatei für Suchmaschinen
 - Adresszeile/Suchfeld: manp<TAB> wget<ENTER>
 - <https://manpages.debian.org/jump?q=wget>
 - <https://manpages.debian.org/jessie/wget/wget.1.en.html>

Laufzeitumgebung

- manziarly ist eine VM auf einem [HP BladeSystem bei Bytemark Hosting](#)
AMD Opteron™ 23xx-System (2008/2009)
2 GB RAM, Storage ist spinning disk
→ Größenordnungen: Minuten (workstation), Stunden (manziarly) ([Messwerte](#))
→ Geschwindigkeit ist wichtig: disaster recovery, developer velocity
- DSA (Debian System Administrators) nutzt ausschließlich Apache2

Laufzeitumgebung: static mirroring

- Debian (und Tor) nutzen ein [static mirroring setup](#) von Peter Palfrader (weasel)
[rsync\(1\)](#) zum Verteilen auf mehrere Server
DNS-Anpassungen vor geplanten Wartungsarbeiten (z.B. Kernel-Upgrades)
- debiman wurde angepasst, um gut damit zu funktionieren
→ `debiman-auxserver` redirects in Apache2-config re-implementiert

Laufzeitumgebung: static mirroring (rewritemap)

```
debiman-idx2rwmap \  
  -index=/srv/manpages.debian.org/www/auxserver.idx \  
  -output_dir=$TMPDIR
```

```
LC_ALL=C sort ${TMPDIR}/output.* > ${TMPDIR}/rwmap.txt
```

```
echo -n
```

```
H4sICEEvilgAA2VtcHR5My5kYm0A7dexCYBADIXhF+GK624DLVxAbJzBMVzBNRzO0tpJPDnlRLEW  
4f8gJCRkgCdJpmRonPw+hFg+7c7bldguqmPv+nmdyrwvj169/AEAAAAAgO+YHnk9mMu3O/I/AAAA  
AAD/sgEYIbKQACAAAA== | base64 -d | gunzip -c > ${TMPDIR}/rwmap.dbm
```

```
/usr/sbin/httxt2dbm -f DB -i ${TMPDIR}/rwmap.txt -o ${TMPDIR}/rwmap.dbm
```

Laufzeitumgebung: static mirroring (apache2)

```
# Replace e.g. pt-BR with pt_BR.
RequestHeader edit Accept-Language "-" "_" early
# Only SetEnvIf gets called _before_ RewriteRule
SetEnvIf Accept-Language "^(.*)$" ACCLANG=$1
SetEnvIf Accept-Language "^$" ACCLANG=en
RewriteEngine on
RewriteMap all dbm:/srv/man/rwmap-all.dbm
    # chomp off the first language tag
    RewriteCond "%{env:ACCLANG}" "^[^,;]+"
    RewriteRule .* - [E=ACCTOK:%1]
    RewriteCond "${all:$1.%{env:ACCTOK}}" "^(.+) $"
    RewriteRule ^(.+) $ /%1 [redirect=307,last]
# while ACCLANG is non-empty, repeat
RewriteCond "%{env:ACCLANG}" "^(?:[^\,;]+), (.+) "
RewriteRule .* - [E=ACCLANG:%1,N]
# fallback: language is already included?
RewriteCond "${all:$1}" "^(.+) $"
RewriteRule ^(.+) $ /%1 [redirect=307,last]
```

Auszug! Vollständige Version:
[80 Zeilen mod_rewrite-Config](#)

Laufzeitumgebung: CDNs/cloud

- Statische Seiten überall einfach zu nutzen, debiman-auxserver einfach zu deployen

```
server {  
    listen 80;  
    root /srv/man;  
    expires 1h;  
    location / {  
        rewrite ^/?$ /index.html;  
        gzip_static always;  
        gunzip on;  
        error_page 404 = @auxserver;  
    }  
  
    location @auxserver {  
        proxy_pass http://localhost:2431;  
    }  
}
```

```
[Unit]  
Description=debiman auxiliary service  
endpoints  
  
[Service]  
Restart=always  
StartLimitInterval=0  
User=nobody  
Group=nogroup  
ExecStart=/usr/bin/debiman-auxserver  
PrivateTmp=true  
ProtectSystem=strict  
  
[Install]  
WantedBy=multi-user.target
```

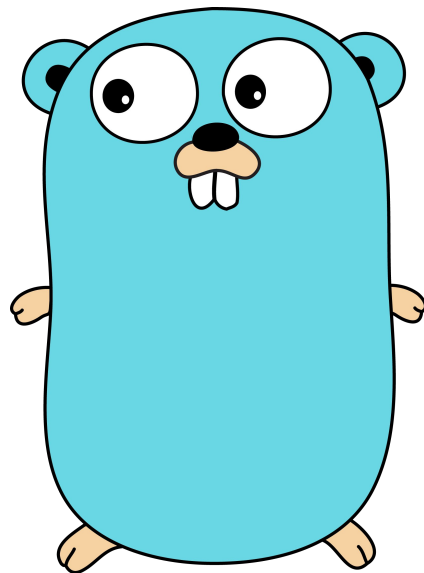

Fazit: manpages.debian.org

- Mobil-Layout für Smartphones, Tablets, etc.

seite/test	crontab.5	gbp.1	javafxpackager.1	zsnes.1	tophat.1	w3m.1.ja	w3m.1.de
m[...].debian	alle 😊	✓	✓	✓	✓	✓	✓
m[...].ubuntu	systemd-cron	✓	✗	✓	✓	❓	✓
man.cx	cron	stable	✗	✓	✗	✓	✗
linux.die.net	cron	✗	✗	✗	✗	✗	✗
mankier.com	cron	✗	✗	✗	✗	✗	✗
man.he.net	BSD	✗	✗	✗	✗	✗	✗
man7.org	cronie	✗	✗	✗	✗	✗	✗

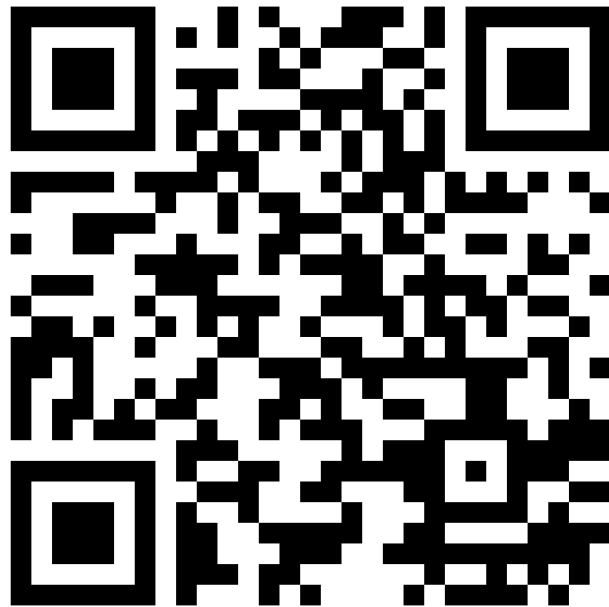
Fazit: Go

- Funktionalität größtenteils implementierbar mit der standard library,
`golang.org/x/{text,net/html}` und `pault.ag/go/debian`
- Nebenläufigkeit einfach umzusetzen
→ maximale Ressourcen-Ausnutzung einfach erreichbar
- Auf `manziarly` (2G RAM) musste man auf Speicher aufpassen
→ [Profilierer pprof hilft](#)



Fragen?

- Mehr Infos auf <https://github.com/Debian/debiman>
- Interesse an einer debiman-Instanz für deine Lieblings-Distro? Sprich mich an!
- Danke für die Aufmerksamkeit! Fragen?
- [Feedback zum Vortrag](#)



(Bonus) Edge case: slave alternative manpages

- Debian-Mechanismus für Wahl einer Implementation (z.B. x-www-browser, vi)
→ nicht nur symlinks für Binaries, auch für manpages
- update-alternatives (8) -Aufrufe im postinst-Script
→ wird zur Laufzeit eingerichtet, keine formelle API
→ wir müssten Pakete installieren (ugh)
- piuparts (1) -Patch zeichnet update-alternatives (8) -Aufrufe auf

(Bonus) Manpages ausliefern: webfont loading

- Browser nutzen FOIT (Flash Of Invisible Text) statt FOUT (Flash Of Unstyled Text)
→ auf langsamen Verbindungen (z.B. 2G) ist 10s lang kein Inhalt zu sehen
- Browser-support für verschiedene Features sehr unterschiedlich:
JavaScript, Font Loading API, Preloading, WOFF/WOFF2, Font Display API
- Details zur Lösung auf <https://michael.stapelberg.de/Artikel/font-loading>